Universidad
Carlos III de Madrid

Escuela Politécnica Superior - Leganés

Departamento de Estadística

PROYECTO FIN DE CARRERA

Ingeniería de Telecomunicación

Reconocimiento de expresiones con Kinect

Autor: Manuel Regidor Serrano
Tutor: David Delgado Gómez

Leganés, abril de 2015

TÍTULO: *Reconocimiento de expresiones con Kinect*
AUTOR: Manuel Regidor Serrano
TUTOR: David Delgado Gómez

EL TRIBUNAL

PRESIDENTE: Ismael Sánchez Rodríguez-Morcillo

VOCAL: Xavier Soldani

SECRETARIO: Ignacio Cascos Fernández

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día 4 de mayo de 2015 en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, el tribunal acuerda otorgarle la CALIFICACIÓN de:

Presidente Secretario Vocal

Agradecimientos

A mis padres, Enrique y Rosario.

A mis hermanos, Carlos y Nico.

A mis abuelos, Enrique, Gabriela, Ángel y Anita.

Al triatlón, deporte que me ha dado tanto en los últimos años de carrera y me ha hecho crecer como persona y deportista.

Al CDE Triatlón Aguaverde, donde he conocido a grandes personas que me han dado fuerzas para no venirme abajo durante estos años.

A mis amigos de toda la vida, Álvaro y Rafa.

A los compañeros y amigos de clase: Dani, Javi, Artem, Adri, Silvia, Jesús, Irene… sin los que posiblemente no hubiera llegado hasta el final.

A mi tutor, David, por ayudarme durante el transcurso de este proyecto.

Tabla de contenido

Tabla de ilustraciones

Resumen

Los progresos recientes en análisis automáticos de rasgos faciales y comportamentales han abierto una puerta a nuevas aplicaciones para la identificación de conductas no verbales. Por ejemplo, en el área de la salud, los algoritmos de visión por ordenador pueden ayudar al personal sanitario a mejorar la comunicación a través de la telemedicina. Los descriptores automáticos del comportamiento pueden además añadir información cuantitativa a las interacciones profesional - paciente. Otro campo de aplicación es el de la psicología, donde una aplicación relevante para este tipo de tecnología no verbal es el desarrollo de descriptores robustos de comportamiento que se correlacionan con trastornos psicológicos como la depresión, ansiedad, trastornos de estrés postraumático o de hiperactividad, etc.

De esta forma, el objetivo de este proyecto fin de carrera ha sido el de detectar variaciones faciales que se presentan en la cara de una persona ante diferentes estímulos. En este caso, los estímulos han consistido en vídeos que se han mostrado a las personas que se sometieron al estudio piloto que se llevó a cabo en la Fundación Jiménez Díaz. Estos estímulos son recogidos a través del nuevo sensor Kinect, disponible en el mercado desde mediados del 2014, y que permite obtener un análisis detallado de los movimientos faciales. Una colección de 3 vídeos de corta duración fue mostrada a los participantes, cada uno de ellos con un objetivo en concreto.

Los resultados del estudio serán analizados en términos de FACS, considerado como uno de los métodos más eficaces para medir los comportamientos faciales. Este sistema divide las expresiones faciales en unidades de acción (UAs) en vez de hacer una clasificación en unas cuantas emociones básicas.

Palabras clave: reconocimiento de expresión facial, FACS, Unidad de Acción, Microsoft Kinect, Trastorno por Déficit de Atención e Hiperactividad.

1. Introducción

El rostro humano es una importante fuente de comunicación que nos permite tanto mostrar nuestras emociones como reforzar nuestra comunicación verbal. A modo de ejemplo, nos permite mostrar interés, desacuerdo o diferentes estados de humor. Incluso permite reconocer el estado anímico de las personas antes de que se produzca la comunicación verbal.

Esta forma de comunicación no verbal la hemos desarrollado a lo largo de miles de años y es especialmente efectiva debido a que existen emociones, llamadas emociones básicas o universales que son iguales en cualquier población. Esta afirmación fue inicialmente sugerida por Darwin [39]. Posteriormente, en 1972, Ekman y Friesen, [5], [6], [7], mostraron que las emociones de alegría, sorpresa, tristeza, miedo, rechazo, odio e ira son básicas o biológicamente universales en la especie humana. Esto difiere de la comunicación corporal o los gestos de la mano que depende de la cultura en la que nos encontremos.

En las últimas décadas se han realizado un gran número de estudios con el objetivo de identificar de manera automática estas expresiones universales a través del análisis de imágenes. Los diversos enfoques que se han explorado incluyen, entre otros, el análisis de los movimientos de las distintas características faciales [25], [26], [27], mediciones de las formas de los rasgos faciales y sus disposiciones espaciales [28] o métodos que relacionan imágenes de la cara con modelos físicos de la musculatura facial [30], [31], [32].

Estos estudios tienen la limitación que son capaces de caracterizar únicamente las citadas seis emociones básicas, dejando aparte cualquier otro tipo de expresión. Por esta razón, en los últimos años, se pretende obtener una descripción más detallada de las emociones que caracterice cualquier posible emoción. Con este objetivo, Ekman, desarrolló el sistema de codificación facial FACS (Facial Action Coding System), que descompone cualquier expresión facial en un conjunto de unidades de acción (UA). Cada una de estas UAs está asociada a un movimiento de uno o varios músculos faciales. Así por ejemplo, la UA 43 se relaciona con cerrar los ojos o la UA 4 con bajar las cejas. De esta forma, además de caracterizar cualquier emoción, podemos tener información sobre variaciones en intensidad dentro de una categoría emocional (enfado frente a furia) o combinar emociones para generar nuevas categorías (felicidad + repulsión = engreimiento). Diversas técnicas han sido propuestas para capturar estas UAs de forma automática [40-43]. Mase [40] y Essa [41] propusieron diversos patrones de flujo óptico que, aunque los autores no lo indicaron, se correspondían con diferentes UAs [46]. Bartlett et al. [42] usaron secuencias de imágenes alineadas manualmente usando tres ejes de coordenadas, rotando las imágenes para que los ojos quedaran en horizontal y finalmente escalando las imágenes a 60x90 pixeles. Su sistema fue entrenado y probado usando un validación cruzada dejando a uno fuera. Estos autores reconocieron seis UAs de la parte superior de la cara (UA 1, UA 2, UA 4, UA 5, UA 6 y UA7) cuando eran presentadas en imágenes de forma individual. Sin embargo, el sistema fallaba en reconocer estas UAs cuando se presentaban en la imagen simultáneamente más de una. Obtuvieron un 90.9% de precisión combinando diferentes tecnologías (entre ellas, el flujo óptico, que es una técnica para estimar el movimiento basada en el principio de conservación de brillo que supone que la intensidad de cada punto entre cada par de frames es constante y que el

desplazamiento del pixel se produce en una pequeña región de la imagen de forma suave). Donato et al. [43] propusieron un sistema similar con la diferencia que en lugar de flujo óptico utilizaron filtros de Gabor. Estos autores obtuvieron una precisión del 95.5% en identificar las seis UAs de la parte superior de la cara anteriormente comentadas, dos de la parte inferior y cuatro combinaciones de UAs, que son tratadas por los investigadores, por sencillez de cálculo, como nuevas UAs.

Un descubrimiento interesante que se obtuvo a través de los estudios del sistema FACS fueron las llamadas microexpresiones consistentes en breves movimientos faciales (de duración inferior a 1/3 de segundo [9]) que se producen de forma involuntaria. La primera vez que aparece el término microexpresión es en el famoso estudio de "Mary" realizado por Ekman [8]. En este trabajo, Ekman, tras analizar una entrevista grabada en vídeo de una paciente con síntomas de depresión y riesgo de suicidio, descubrió que realizaba pequeños movimientos con el hombro y microexpresiones asociadas al labio justo antes de mentir a una pregunta.

Este proyecto, persigue identificar de forma automática que UAs se activan, así como su intensidad, cuando se presentan determinados estímulos a través de vídeos. Para ello, se utilizará la nueva cámara Kinect, que proporciona diferentes índices similares a las UAs durante cada uno de los instantes que dura el vídeo.

El resto de la memoria se estructura como se explica a continuación.

En el capítulo 2 se presenta una breve revisión bibliográfica y estado del arte de la identificación y análisis de expresiones faciales.

El capítulo 3 muestra el sistema FACS que permite describir la actividad facial en términos de músculos faciales.

El capítulo 4 se centra en Kinect. Tras realizar una breve exposición sobre su historia, se muestran sus principales características tanto de hardware como de software. En este capítulo también se muestra la relación entre diversos índices proporcionados por el Kit de desarrollo de Kinect y las distintas UAs.

En el capítulo 5 se expone el procedimiento utilizado para el desarrollo de este proyecto, partiendo de la implementación software hasta el escenario utilizado para la obtención de muestras.

En el capítulo 6 se muestran los resultados obtenidos.

La memoria acaba en el capítulo 7 con las conclusiones y futuras líneas de investigación.

Además, en el anexo I se muestran las diferentes fases de desarrollo; y en el anexo II, el presupuesto y coste del proyecto.

2. Revisión bibliográfica

Desde que Microsoft lanzara el *software development kit* (SDK) para *Kinect for Windows* en junio de 2011 que permite a los usuarios crear sus propias aplicaciones, muchos investigadores han visto en esta herramienta un gran potencial. A modo de ejemplo, ha sido utilizada para enseñar anatomía (https://www.youtube.com/watch?v=jORsG8AG72I), crear modelos 3D (https://www.youtube.com/watch?v=V1Nsmn-xkKI) , desarrollar interfaces naturales de usuario (https://www.youtube.com/watch?v=ShyNV9hHPpo) o incluso para dirigir a personas invidentes en la realización de ejercicios de yoga (https://www.youtube.com/watch?v=cm_ghJPqj70), ver imagen Figura 1.

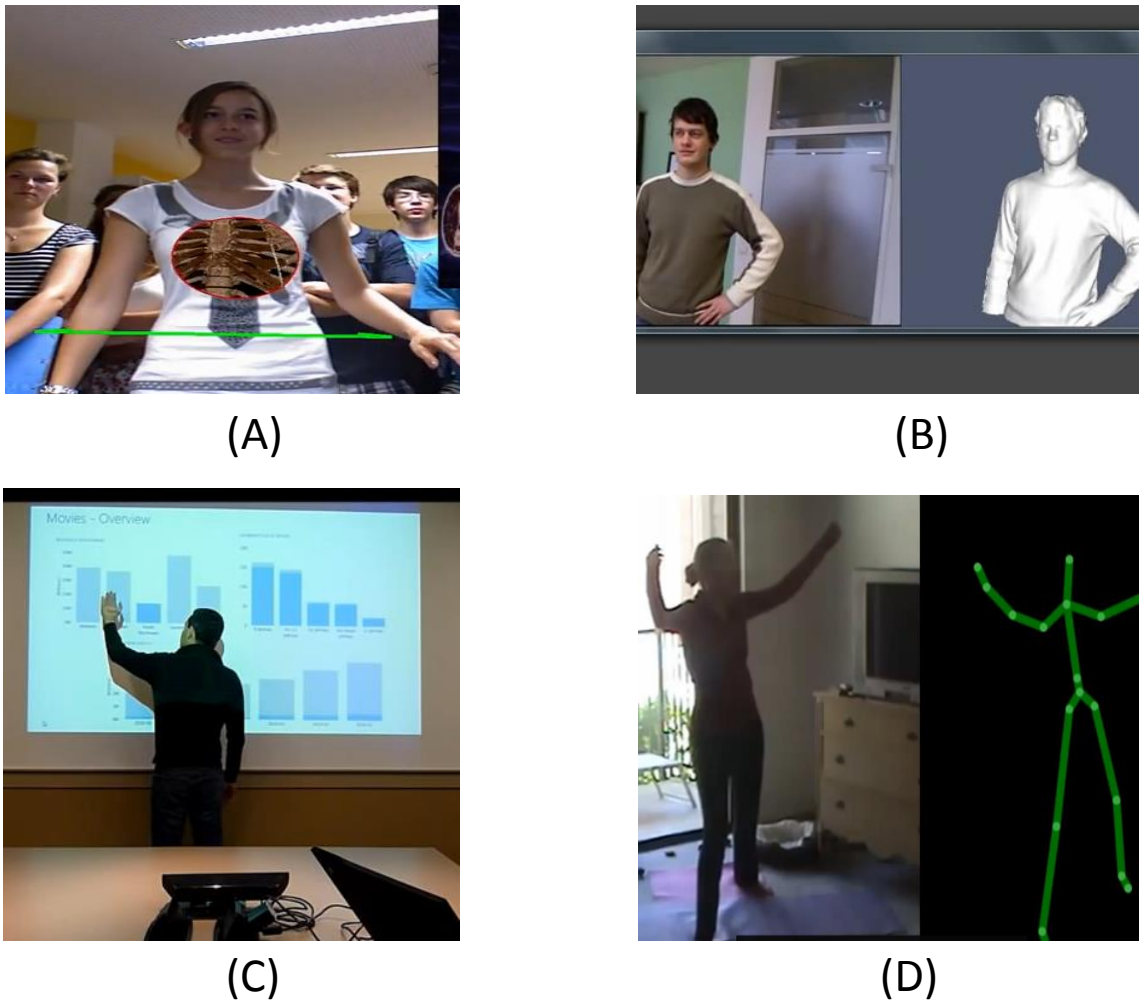


Figura 1: Ejemplos de aplicaciones utilizando la cámara Kinect. A: Enseñanza de Anatomía. B: Modelado 3D. C: Interfaz Natural de Usuario. D: Enseñanza de Yoga.

2.1 Literatura relacionada con reconocimiento de emociones y Kinect

A pesar de que existen gran cantidad de trabajos orientados al reconocimiento de expresiones, aún no se han realizado muchos estudios utilizando la cámara Kinect. Entre los pocos estudios encontrados, se encuentra el llevado a cabo por Adam Wyrembelski [1], que propone un sistema basado en Kinect para reconocer las seis expresiones básicas mediante la detección de determinados UAs y un algoritmo de K vecinos. Sin embargo, el autor no realiza ningún experimento y por tanto es difícil valorar la adecuación de su propuesta. De forma similar, Vineetha, Sreeji y Lentin [2], propusieron una aplicación para detectar emociones utilizando la cámara Kinect utilizando diversas variables explicativas obtenidas mediante técnicas de segmentación de visión por ordenador y redes neuronales como clasificador. Desafortunadamente, al igual que el trabajo anterior, no desarrollaron ningún experimento y por tanto es difícil valorar la viabilidad de su propuesta.

Un estudio más elaborado es el realizado por Stocci [47]. En este trabajo, el autor muestra que utilizando cinco unidades de animación (a partir de ahora, referidas con K-UAs) es posible obtener una precisión cercana al 95% en el reconocimiento de las expresiones de alegría, miedo y sorpresa y de un 70% de las expresiones de tristeza y enfado. Las cinco unidades de acción que se utilizaron fueron la apertura de mandíbula, estiramiento de labios, movimiento hacia abajo de la comisura de los labios, movimientos hacia abajo o hacia fuera de las cejas. El inconveniente de este trabajo es que únicamente se utilizaron datos provenientes de seis personas y no se describe como fue realizado el experimento.

Un artículo más interesante, aunque utiliza análisis corporal y no facial, es el desarrollado por varios autores de la universidad de Génova [36]. Estos proponen un método para reconocer estados de ánimo a partir de movimientos del cuerpo y gestos con un clasificador Support Vector Machine (SVM) con el objetivo de intentar ayudar a niños con autismo. Para ello, el sistema propuesto detecta automáticamente la posición de diversos puntos corporales como las manos, hombros o cabeza. Estos autores evalúan su sistema en una muestra de 100 videos grabados por 12 actores que realizan gestos y movimientos relacionados con las seis emociones básicas. Los autores indican que su sistema es capaz de identificar correctamente un promedio de 74.7% de las expresiones de felicidad cuando no se incluyen en el análisis los videos que indican odio o la sorpresa y un 59.8% cuando consideran las seis emociones básicas. Estos números fueron obtenidos mediante 50 validaciones cruzadas utilizando conjuntos de entrenamiento/test. Los autores reportaron números ligeramente inferiores cuando se utiliza validación cruzada dejando uno fuera. Además, se señala que los resultados obtenidos son muy parecidos a los obtenidos con una validación humana (60 personas anónimas que no participaron en el proceso de obtención de los datos).

Un aspecto que es importante resaltar es la creación de bases de datos, como producto derivado de estos estudios, con las que se puedan realizar investigaciones futuras en esta área. A modo de ejemplo, la base de datos FEEDB [37] contiene grabaciones de personas mostrando emociones obtenidas a través del sensor Kinect. Incluye 1650 grabaciones de 50 personas mostrando 33 expresiones faciales y emociones. Todas las grabaciones son clasificadas por expertos en la escala de PAD (PAD, *Pleasure, Arousal, Dominance* que es una escala psicológica

desarrollada por Mehrabian y Russell que describe y mide estados emocionales usando tres valores numéricos para estudiar la comunicación no verbal) y FACS. Además, se añaden metadatos como la edad, sexo, resolución, postura de la persona, orientación, etc. En la creación de esta base de datos, el autor señala que no es fácil obtener ciertas expresiones ya que muchas personas tienen ciertos problemas en expresar emociones en un entorno controlado (bajo demanda) así como que hay importantes diferencias expresivas por diferentes personas (por ejemplo debido a una cultura o sociedad determinada). Finalmente, se apunta, que esta base de datos será mejorada en un futuro añadiendo expresiones naturales, y no solo fijas, obtenidas a través del comportamiento ante videojuegos. Los autores ofrecen esta base de datos de forma gratuita a la comunidad.

Otra base de datos de expresiones faciales en 3D que también es importante destacar es FaceWarehouse [50]. Esta base de datos contiene la información RGBD de Kinect de 150 individuos con diversos orígenes étnicos y edades comprendidas entre los 7 y 80 años. Para cada persona, se captura su expresión neutra junto con otras 19 como puede ser, a modo de ejemplo, la apertura de la boca o la sonrisa. Además, contiene la posición de diversos puntos faciales. De esta forma, según los autores, es posible representar la mayoría de las expresiones faciales. Finalmente, muestran la importancia de esta base de datos utilizándola en una amplia variedad de aplicaciones tales como la manipulación de imágenes faciales, la transferencia de componentes de la cara en tiempo real, etc.

Concluimos esta sección indicando otra base de datos relevante que es la que creada por Høg et al. de la universidad de Aalborg [51]. Esta base de datos contiene información obtenida de 31 personas en 17 poses y expresiones faciales (en total, 1581 imágenes). La forma que utilizaron los autores para capturar las diferentes posiciones de la cara fue la de establecer 13 puntos en la pared detrás del sensor Kinect para que cada persona mirara secuencialmente a los números y obtener los mismos ángulos para cada individuo. Sin embargo, esta base de datos no está orientada al reconocimiento de expresiones si no a la detección y reconocimiento de personas bajo diferentes expresiones. Los autores señalan que, su sistema es capaz de detectar cuando una cara se encuentra enfrente de la Kinect en un 93.62% de ocasiones.

2.2 Literatura relacionada con reconocimiento de microexpresiones

Aunque el reconocimiento de microexpresiones no es un objetivo de esta memoria, por completitud de los apartados anteriores, haremos una breve mención de este tipo de expresiones.

Tal y como se ha explicado en la introducción, es conocido que una expresión facial como tal (macroexpresiones), no se suele dar en la mayoría de las ocasiones. Las macroexpresiones son las que ocurren en múltiples regiones de la cara y son fácilmente observables (expresiones universales) mientras que las microexpresiones, son las que ocurren rápido y en pequeñas regiones de las caras, más difíciles de detectar para el ojo humano. Según Ekman, cuando una persona trata de ocultar sus sentimientos, las emociones verdaderas se filtran de forma rápida

e incontrolada y pueden manifestarse como microexpresiones o expresiones de corta duración (Ekman & Friesen, 1969).

Entre los trabajos aparecidos, podemos citar el desarrollad por Shreve et al. [10]. En este trabajo, los autores dividieron la cara en regiones (cara, mejillas, frente y ojos) y calcularon la deformación facial para cada sub región. El patrón de deformación en cada sub región es después analizado para detectar microexpresiones. En otro trabajo, Pfister et al. [11] propusieron un marco de trabajo usando interpolación temporal y múltiples kernels para aprender a reconocer microexpresiones. Polikovsky et al [12], usaron un gradiente en 3D como descriptor para el reconocimiento de las microexpresiones. Wu et al. [13] desarrollaron un sistema automático de reconocimiento de microexpresiones usando máquinas de vector de soporte, SVMs por sus siglas en inglés. Wang et al. [14] en cambio utilizaron técnicas de "Extreme Learning Machine" (ELM) para el reconocimiento de las microexpresiones.

Por el contrario, hay muy pocas bases de datos de microexpresiones bien desarrolladas, lo que afecta al desarrollo de la investigación en este campo. La obtención de microexpresiones es difícil, ya que solo se presentan en ciertas situaciones en las que un individuo intenta suprimir las emociones que siente, pero falla al hacerlo. Algunos investigadores han desarrollado pequeñas bases de datos de microexpresiones pidiendo a los participantes que establecieran expresiones faciales rápidamente. Sin embargo, el resultado obtenido es diferente al que ocurriría con microexpresiones obtenidas de forma espontánea.

Se ha visto que viendo videos de carácter emotivo a la vez que se intenta neutralizar los sentimientos que ese vídeo produce es un método eficaz para obtener microexpresiones espontáneas sin muchos movimientos faciales irrelevantes [20]. Hasta ahora, según nuestra búsqueda, solo hay cuatro bases de datos de microexpresiones disponibles de manera pública y solo dos de ellas contienen microexpresiones espontáneas. Estas son USF-HD,Polikovsky's database,SMIC database y CASME database, siendo estas dos últimas las que contiene expresiones espontáneas. Un resumen de las características de estas bases de datos puede encontrarse en [21], "*CASME II: An Improved Spontaneous Micro-Expression Database and the Baseline Evaluation*". En dicho artículo, se propone el uso de una nueva base de datos que mejora la anterior (CASME I).

Todos estos estudios han mostrado los potenciales del análisis facial para obtener descriptores automáticos y lo mucho que queda por descubrir. Como se irá mostrando a lo largo de la memoria, Kinect posee gran potencia en esta tarea al automatizar gran parte del trabajo que se venía realizando anteriormente: detección de puntos faciales, seguimiento de estos puntos a lo largo del tiempo, detector de emociones, etc.

3. Facial Action Coding System (FACS)

En vez de clasificar directamente las expresiones faciales en un número finito de emociones básicas, también es posible reconocer los músculos faciales que subyacen a los movimientos para después interpretarlos en términos de categorías como emociones, actitudes o estados de ánimo.

El *Facial Action Coding System (FACS),* es el más conocido, común y amplio sistema utilizado por observadores humanos altamente entrenados para describir la actividad facial en términos de acciones de músculos faciales, ya que permite medir de forma cuantitativa movimientos faciales.

FACS fue desarrollado en 1978 por Ekman & Friesen a través del estudio de la cara mediante técnicas de un alto grado de conocimiento en anatomía humana, palpación y estudio. Ekman & Friesen descubrieron como la contracción y la distensión de los músculos faciales altera la apariencia facial. Basado en este hecho, comenzaron a estudiar todas las diferentes expresiones faciales que una persona era capaz de realizar y categorizaron todas las acciones fundamentales de músculos individuales o grupos de ellos en 44 categorías llamadas las Action Units (UAs, unidades de acción, en castellano): 30 UAs son anatómicamente relacionadas con las contracciones de músculos faciales específicos: 12 para la parte superior de la cara y 18 para la parte inferior. Las UAs pueden ocurrir individualmente o como una combinación de ellas. De este modo, aunque el número de las UAs es relativamente pequeño, más de 7000 combinaciones de estas unidades de acción se han observado [22].

Sin embargo, FACS fue un sistema diseñado para realizar un análisis fotograma a fotograma, y por lo tanto, inútil para test en tiempo real. Por poner un ejemplo, codificar dos minutos de vídeo por un experto en FACS, puede llevar más de dos horas de trabajo, tal y como nos explica Ryan et al. en “Automated Facil Expression Recognition System”.

Por todo lo anterior, el mayor impedimento para su despliegue es el tiempo requerido para entrenar personas expertas y para manualmente puntuar el vídeo fotograma a fotograma. Para alcanzar unos resultados mínimos en FACS, cada minuto de vídeo que se quiere analizar en busca de UAs lleva aproximadamente una hora de promedio. Por tanto, como parece evidente, un sistema automático haría este sistema mucho más accesible como herramienta para la investigación y valoración del comportamiento al mejorar la exactitud, precisión y resolución temporal de las medidas faciales. Otra ventaja de utilizar este sistema está clara desde el primer momento. Usando las UAs, es posible referirse a todas las expresiones faciales de forma inequívoca y determinar fácilmente la emoción de usuario.

Friesen y Ekman también han desarrollado EMFACS (Emotional FACS), que usa los mismos principios que FACS con la diferencia de centrarse exclusivamente en ciertas emociones. De esta manera, el abanico de UAs a detectar es mucho menor, con lo que el coste computacional es significativamente inferior, lo que nos permite realizar un análisis de expresiones en tiempo real.

De este modo, desarrollando sistemas que sea capaces de reconocer UAs, seremos capaces en un futuro de tener sistemas de análisis en tiempo real que funcionarán, por ejemplo, demostrando diferencias entre un dolor genuino y otro simulado, diferencias entre decir la verdad y mentir, diferencias entre signos faciales que puedan indicar tendencias suicidas en pacientes deprimidos…

Sabiendo esto, nuestro primer paso para desarrollar este sistema es el de analizar como FACS (*Facial Action Coding System*) funciona y como las UAs (*FACS Unidades de Acción*) se relacionan con las *Animation Units (K-UAs)* que nos ofrece Kinect.

En la Tabla 1 se puede ver un resumen de las Unidades de Acción anteriormente comentadas, así como los músculos implicados. Además, en http://www.cs.cmu.edu/~face/facs.htm, se puede apreciar visualmente los movimientos que se generan con cada activación de diferentes UAs.

NÚMERO UA	NOMBRE FACS	MÚSCULOS IMPLICADOS
0	Neutral face	
1	Inner Brow Raiser	frontalis (pars medialis)
2	Outer Brow Raiser	frontalis (pars lateralis)
4	Brow Lowerer	depressor glabellae, depressor supercilii, corrugator supercilii
5	Upper Lid Raiser	levator palpebrae superioris, superior tarsal muscle
6	Cheek Raiser	orbicularis oculi (pars orbitalis)
7	Lid Tightener	orbicularis oculi (pars palpebralis)
8	Lips Toward Each Other	orbicularis oris
9	Nose Wrinkler	levator labii superioris alaeque nasi
10	Upper Lip Raiser	levator labii superioris, caput infraorbitalis
11	Nasolabial Deepener	zygomaticus minor
12	Lip Corner Puller	zygomaticus major
13	Sharp Lip Puller	levator anguli oris (also known as caninus)
14	Dimpler	buccinator
15	Lip Corner Depressor	depressor anguli oris (also known as triangularis)
16	Lower Lip Depressor	depressor labii inferioris
17	Chin Raiser	mentalis
18	Lip Pucker	incisivii labii superioris and incisivii labii inferioris
19	Tongue Show	
20	Lip Stretcher	risorius w/ platysma
21	Neck Tightener	platysma
22	Lip Funneler	orbicularis oris
23	Lip Tightener	orbicularis oris
24	Lip Pressor	orbicularis oris
25	Lips Part	depressor labii inferioris, or relaxation of mentalis or orbicularis oris
26	Jaw Drop	masseter; relaxed temporalis and internal pterygoid
27	Mouth Stretch	pterygoids, digastric
28	Lip Suck	orbicularis oris
29	Jaw Thrust	
30	Jaw Sideways	
31	Jaw Clencher	masseter
32	[Lip] Bite	
33	[Cheek] Blow	
34	[Cheek] Puff	
35	[Cheek] Suck	
36	[Tongue] Bulge	
37	Lip Wipe	
38	Nostril Dilator	nasalis (pars alaris)
39	Nostril Compressor	nasalis (pars transversa) and depressor septi nasi

41	Glabella Lowerer	Separate Strand of AU 4: depressor glabellae (aka procerus)
42	Inner Eyebrow Lowerer	Separate Strand of AU 4: depressor supercilii
43	Eyes Closed	Relaxation of levator palpebrae superioris
44	Eyebrow Gatherer	Separate Strand of AU 4: corrugator supercilli
45	Blink	Relaxation of levator palpebrae superioris; contraction of orbicularis oculi (pars palpebralis)
46	Wink	orbicularis oculi

Tabla 1. Lista de Action Units (FACS). Fuente: Wikipedia

4. Kinect

En este capítulo, se describe inicialmente la historia de proyectos que han dado lugar a la creación de la cámara Kinect. Entenderemos que, a pesar de su bajo coste, la cámara Kinect es más que un simple dispositivo para jugar, con un desarrollo de casi 30 años que incluye entre otros elementos, algoritmos de aprendizaje máquina y varias patentes. Por tanto, tras una breve historia (4.1), se realizará una breve descripción de los diferentes componentes hardware de los que disponemos en la cámara (4.2) y que datos nos ofrece. En la siguiente sección, 4.3, se hablará sobre las características del SDK que Microsoft ha desarrollado para esta versión de la Kinect (v2). Finalmente, en 4.4, ahondaremos en el SDK para ver qué información nos ofrece y cómo podemos utilizarla.

4.1 Historia

Los comienzos de Kinect se remontan a los años 70 en el *Massachusetts Institute of Technology* (MIT), cuando Chris Schmandt comandó un proyecto llamado "Put-that-there", cuyo objetivo era que un usuario moviera objetos que se mostraban en una pantalla. Para ello, utilizaba instrucciones dadas mediante voz y gestos. Los gestos se reconocían mediante un cubo magnético en forma de pulsera magnética y la voz a través de un micrófono. Un esquema y una fotografía del escenario en el que se realizó este proyecto pueden apreciarse en la Ilustración 1.

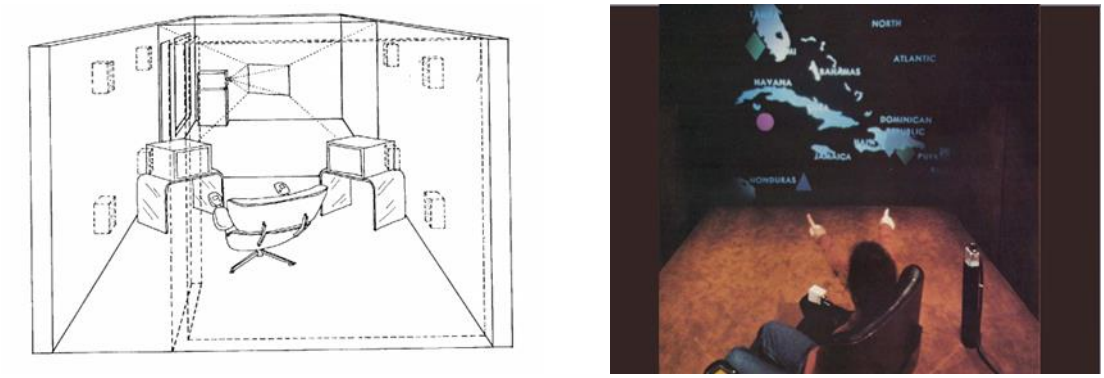


Ilustración 1: Esquema y Fotografía del proyecto "Put-That-There"

En los años posteriores, se desarrollaron proyectos que proporcionaron pequeñas mejoras, pero que sin embargo, no añadieron grandes diferencias. No fue hasta finales de los años 90 cuando se desarrolló la primera interfaz de usuario avanzada que utilizaba un sistema de visión para el reconocimiento de los gestos del usuario. Esta interfaz, llamada "DreamSpace", fue desarrollada por Mark Lucente para IBM, que la propuso como alternativa a los dispositivos

habituales de entrada: teclado y ratón. La Ilustración 2 muestra un esquema y una fotografía del proyecto “DreamSpace”.

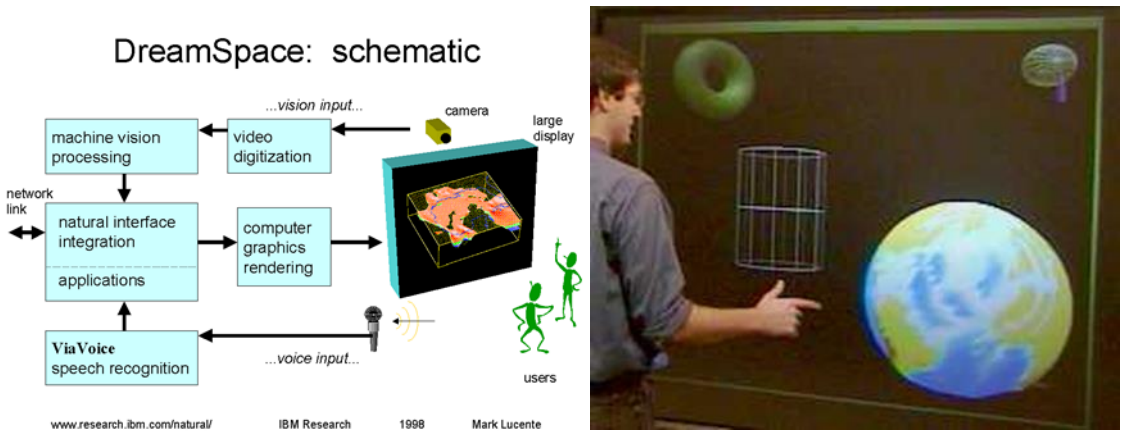


Ilustración 2: Esquema y Fotografía del proyecto “DreamSpace"

Una curiosidad que no podemos dejar pasar en el desarrollo de la Kinect, es la colaboración de John Underkoffler, socio de Lucente, en la película “Minority Report” (2002). Esta película, junto con algunas similares de décadas anteriores, como “StartTrek” (1979), aumentó de forma considerable el interés por las interfaces de lenguaje natural. Pocos años después del estreno de “Minority Report”, en el 2006, Nintendo lanzó al mercado la consola Wii y el controlador remoto (ver Ilustración 3).

Ilustración 3: Controlador remoto Wii.

El dispositivo de Nintendo podía detecta el movimiento de la mano del usuario a lo largo de los tres ejes de la misma manera que se hacía con el cubo magnético del “Put-That-There”. La aparición de esta consola y su dispositivo remoto situó a Nintendo en la cabeza de empresas del sector de videojuegos.

La rivalidad existente entre las distintas compañías de videojuegos provocó que Microsoft iniciase dos proyectos en paralelo en 2006 con el objetivo de desbancar a la consola de

Nintendo. Estos dos proyectos fueron encargados a dos compañías diferentes: PrimeSense y 3DV. El objetivo era presentar un prototipo en la conferencia E3-2007. A pesar del esfuerzo, no consiguieron tener algo para mostrar en esta conferencia. En 2008, Don A. Matrick retoma el proyecto y en 2009 se inicia el "proyecto Natal" dirigido por Alex Kipman que posteriormente concluiría con el nacimiento de la cámara Kinect. Este proyecto, perseguía desarrollar un dispositivo que realizara reconocimiento basado en profundidad, seguimiento de movimiento, reconocimiento de voz y facial con una fecha límite de finales del 2010 (Kinect llegó al mercado en noviembre de 2010). Para ello, tomaron como punto de partida el trabajo realizado por PrimeSense, el cual era capaz de procesar datos de profundidad a 30fps gracias al desarrollo de un novedoso algoritmo basado en un patrón de puntos infrarrojos. El trabajo desarrollado por 3DV fue olvidado y la compañía fue comprada, como ocurre en muchos casos, para evitar problemas de patentes. Microsoft añadió posteriormente un array de cuatro micrófonos con el objetivo de implementar un reconocedor de voz y permitir la cancelación de ecos.

En este punto, el hardware de la Kinect había sido prácticamente completado, pero sin embargo, no el desarrollo software. Para solucionar estas dificultades, se incluyeron en el proyecto, grupos de diversa índole de Microsoft Research (MR). El desarrollo del software encargado de la detección del individuo y de su seguimiento fue desarrollado conjuntamente en los centros de MR de Cambridge y Mountain View utilizando técnicas de aprendizaje máquina. En particular, se utilizaron árboles y bosques de decisión sobre una cantidad enorme de videos grabados que contenían multitud de personas realizando tareas tanto cotidianas como específicas. Por otro lado, MR en Redmond, se encargó del desarrollo del modelo auditivo. Para ello, filtraron ruidos utilizando diversas patentes que son proporcionadas de manera gratuita en el Kinect SDK para Windows. Tras ello, utilizando también técnicas de aprendizaje máquina se desarrolló el modelo que actualmente ha sido incluido en la característica *Tellme* de la Xbox y en el *Kinect for Windows: Runtime Language Pack*. Esta fase fue completada el 26 de septiembre del 2010. El 4 de noviembre del 2010, la primera versión de la Kinect fue comercializada.

En julio de 2014, llegó la segunda versión de la Kinect (Ilustración 4: Kinect for Windows v2), que había sido anunciada un año antes, con importantes mejoras que serán comentadas en el siguiente apartado. Esta nueva versión será la utilizada en este proyecto y por tanto, cuando mencionemos Kinect, haremos referencia siempre a este modelo. Más información de las características de esta versión se recogen en [44] y en Human Interface Guidelines (HIG). Es importante destacar como ya veremos, que debido a las mejoras introducidas en esta versión, los requisitos para poder utilizarla en un PC son bastante elevados.

Ilustración 4: Kinect for Windows v2

4.2 Características, hardware y requisitos de *Kinect for Windows v2*.

La cámara Kinect v2 es básicamente un dispositivo que cuenta con una cámara RGB, un conjunto de micrófonos, un sensor de profundidad y un sensor de infrarrojos. Esto, unido a un poderoso SDK en la que Microsoft ha puesto bastantes recursos nos permite, entre otras:

- Reconocer si una persona está enfrente del sensor y en qué posición se encuentra su cuerpo (sentado, de pie, mirando a la cámara, etc).
- Capturar audio de forma direccional (± 50 grados enfrente del sensor) evitando ruidos gracias a un array de 4 micrófonos. Estos nos permiten cancelar hasta 20 decibelios del ruido ambiente. Además, gracias a la disposición de los micrófonos, es posible detectar la ubicación y movimiento de las personas. Por defecto, Kinect se fija en la entrada de audio más potente.
- Reconocimientos del habla.
- Usar numerosos plugins de terceros, como Unity.
- Gran campo de visión (verticalmente 60 grados y horizontalmente, 70) para la profundidad y el color (objetivo gran angular). Este rango abarca desde el medio metro hasta los 4.5 metros (8 metros con la visión de profundidad). Sin embargo, el punto óptimo para la detección de cuerpos se sitúa entre los 0.8m y los 3.5m.
- Cámara de color (RGB, red-green-blue) de alta definición con resolución 1920x1080 pixeles, trabajando a 30fps o 15fps, dependiendo de la exposición, es decir, de la iluminación de la escena. En escenas oscuras se trabajará a 15fps con el fin de que entre captura y captura pueda entrar más luz al sensor.
- Cámara infrarroja (IR) independiente del color (siempre funcionando a 30fps). La iluminación de la escena está asegurada con 3 sensores IR. La cámara IR es utilizada además para la adquisición de los puntos de profundidad (depth map), aparte de la información IR, con una resolución de 512x242 pixeles. En Ilustración 5 se muestra una captura de datos con cada uno de estos sensores mencionados anteriormente.
- 25 puntos esqueléticos disponibles para seguimiento de hasta seis personas (los puntos se concentran en caderas, hombros y columna vertebral). También se ofrecen puntos individuales (como el pie) así como un continuo seguimiento ante rotaciones de las distintas partes del cuerpo. En el caso de que una persona se encuentre sentada, solo se ofrecen 10 puntos de la parte superior.
- Seguimiento del dedo pulgar y estados de la mano: abierta / cerrada

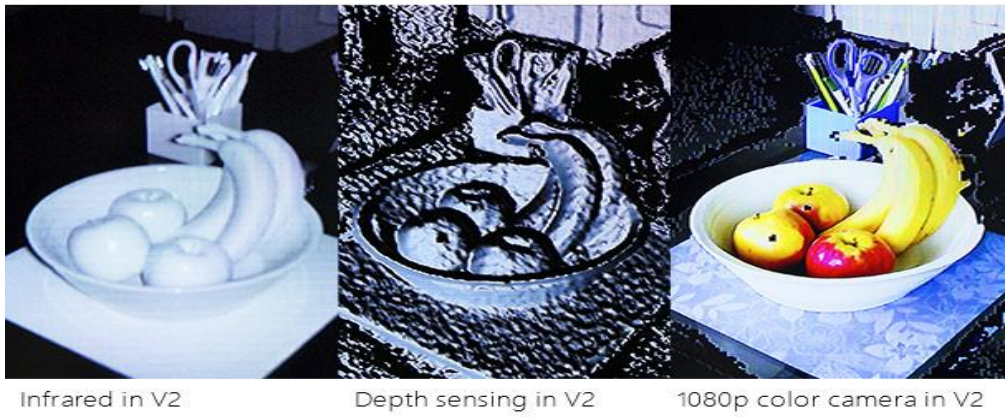


Ilustración 5: Resultados obtenidos con Kinect por los diferentes sensores [44].

La Ilustración 6 recoge la ubicación de los sensores del dispositivo Kinect, tanto en una visión frontal (izquierda), como en una visión sin la carcasa exterior que deja a la vista la ubicación de los sensores.

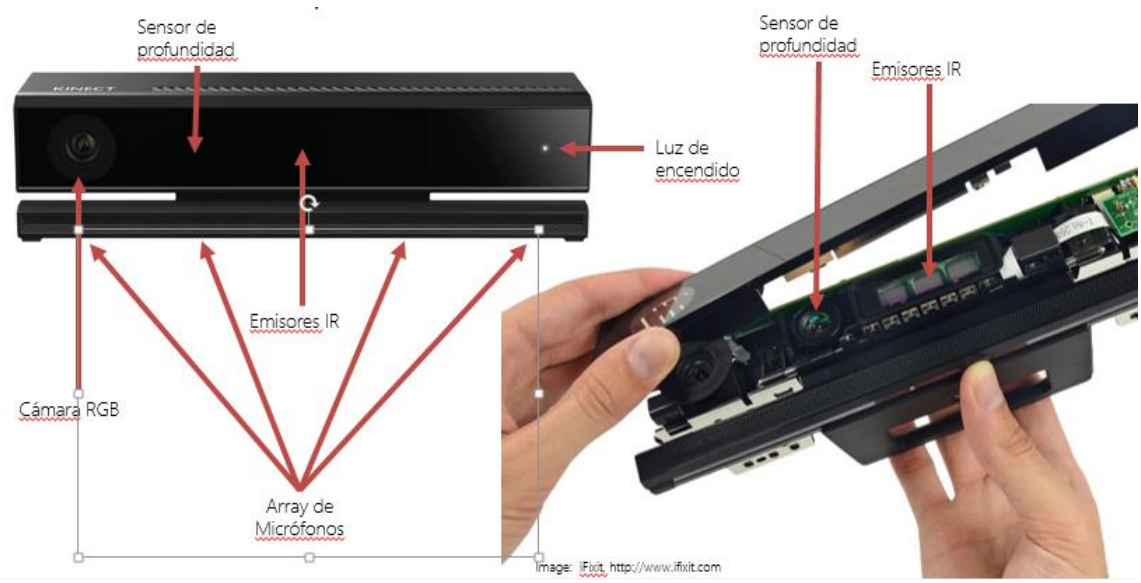


Ilustración 6. Ubicación Hardware Kinect. Fuente ifixit.com

Es importante citar además las diferentes formas que tenemos de combinar los datos que nos ofrecen los sensores de la cámara Kinect para conseguir varias fuentes de información.

- Infrarrojos (IR)
- Profundidad (RGB-D)
- BodyIndex (seguimiento de diferentes personas)
- Body (detección de diferentes partes del cuerpo)
- Audio
- Color (RGB)

En la Ilustración 7, se muestra un fotograma de cada una de estas características.

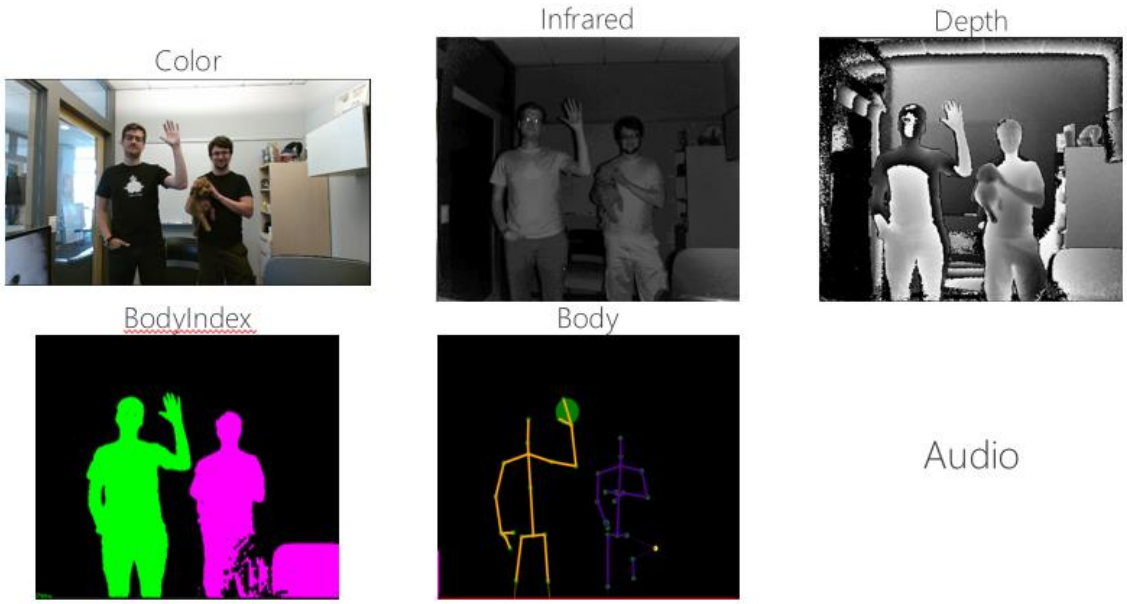


Ilustración 7. Origen de datos en Kinect [48].

Debido a todas las novedades que incluye esta versión de Kinect v2, el hardware necesario para poder ejecutar las aplicaciones sin problemas ni cuellos de botella es bastante exigente con el fin de mantener una velocidad de fotogramas óptima. Microsoft, nos recomienda, a través de [44], las siguientes especificaciones mínimas.

- Procesador de 64 bits (x64).
- Doble núcleo físico de 3.1 GHz o mejor.
- Controlador dedicado de USB 3.0. Este es un requisito importante, pues con USB 2.0 no nos funcionará la *Kinect for Windows v2* debido a que el ancho de banda utilizado es muy alto y con USB 2.0 no se alcanzan las velocidades de transferencia adecuadas. Es fácil identificar los conectores USB 3.0 de los nuevos equipos que están a la venta pues son conectores azules que difieren de los USB 2.0, negros. En el supuesto de que un ordenador de sobremesa no disponga de USB 3.0, este accesorio se puede añadir con tarjetas PCI Express (en el caso de tener slots disponibles).
- 4 GB de RAM
- Tarjeta gráfica que soporte DirectX 11. Este es otro requisito indispensable, pues no conseguiremos ejecutar Kinect v2 en un ordenador que disponga de una tarjeta gráfica con DirectX 10.1 o menor.
- Windows 8, 8.1 o Windows Embebido 8.

Además, Microsoft nos proporciona una herramienta para asegurarnos de que nuestro PC o tableta cumple con los requisitos de compatibilidad con el hardware y el software de Kinect. Esta se puede encontrar en [45].

4.3 Kinect SDK

La versión anterior de la Kinect estuvo marcada por la coexistencia de varias librerías no oficiales junto con el SDK proporcionado por Microsoft. Esto fue debido principalmente a que Microsoft no puso demasiado énfasis en el apartado software tanto como lo hizo en el hardware, lo que hizo que surgieran alternativas que mitigaran la falta de características. Sin embargo, en la v2, esto ha cambiado, y actualmente solo se dispone de las librerías proporcionadas por el equipo de Microsoft.

El Kinect SDK proporciona los drivers y un conjunto de librerías que permiten el desarrollo de aplicaciones que utilizan Kinect como dispositivo de entrada. La interfaz de usuario natural (NUI, Natural User Interface) es el núcleo de la Kinect para la Windows API que nos permite obtener los diferentes flujos de datos que nos ofrecen los sensores de los que dispone Kinect. En la Ilustración 8 se muestra como las aplicaciones interactúan con los flujos de datos y en la Ilustración 9 se muestra la arquitectura del SDK. Además, junto con el SDK, se proporciona un Developer Toolkit que contiene diferentes programas realizados utilizando dicho SDK. Este kit de desarrollo permite realizar aplicaciones en C++, C#, VB o HTML corriendo sobre Windows 8 o Windows 8.1, tanto en arquitecturas x86 o x64.

Algunos de los programas proporcionados pueden servir como prueba y como punto de partida para aplicaciones que se quieran desarrollar. Un ejemplo podría ser el Depth Basics-WPF, que demuestra cómo utilizar DepthFrameReader para obtener y visualizar frames de profundidad. Un *frame* contiene los datos entregados desde el sensor asignados de forma temporal con el fin de evitar la adjudicación permanente de memoria. De este modo, una aplicación debe obtener los datos de cada frame y después cerrar o desechar este objeto tan rápido como sea posible para liberar el identificador subyacente. De no respetar este procedimiento, al estar recibiendo frames de forma continua (una porción de memoria para cada uno), no recibiríamos más frames con el fin de evitar el colapso de la máquina por no disponer de recursos suficientes. Recordemos que son varios los frames que se reciben por segundo para cada flujo de datos.

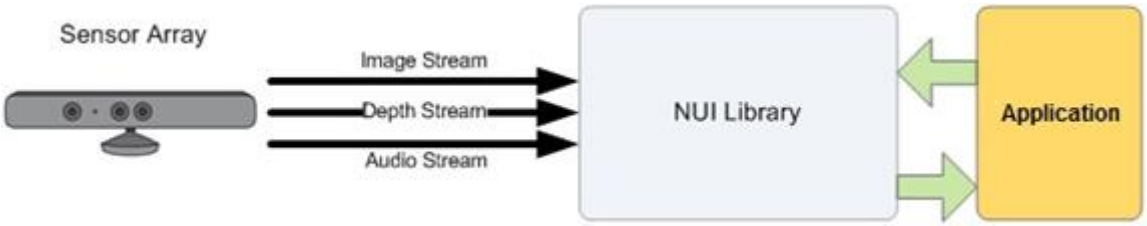


Ilustración 8. Interacción de aplicación con Kinect [38]

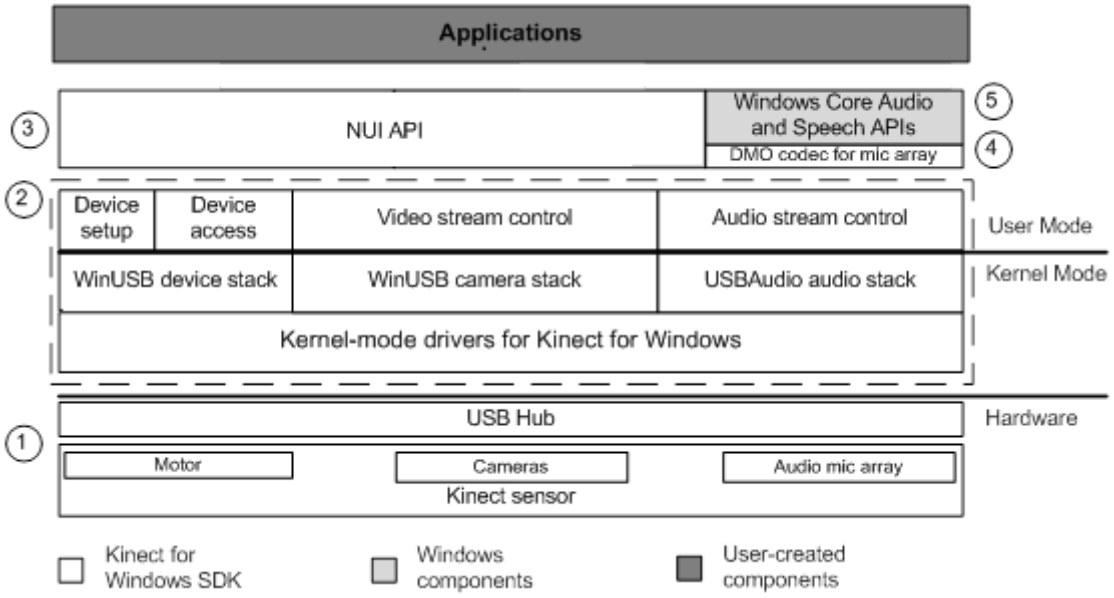


Ilustración 9. Arquitectura del Kinect SDK [38]

Además, el Developer Toolkit proporcionado contiene una aplicación llamada "Kinect Studio" que permite la grabación de los diferentes flujos de datos que nos proporcionan los sensores de la Kinect, además del Visual Gesture Builder (VGB), que nos permite generar datos que se utilizan para realizar detección de gestos y el seguimiento de éstos en tiempo de ejecución.

Finalmente, un vídeo recopilatorio que nos muestra las capacidades del SDK y la nueva versión de Kinect es el siguiente: https://www.youtube.com/watch?v=i9906y10GzQ. Como se puede ver, entre las diversas áreas de aplicación están la medicina (herramienta de apoyo a cirujanos, consultas de médico), la robótica, la educación o el entretenimiento (videojuegos, deporte), como se nos muestra en la Ilustración 10.

Ilustración 10. Ejemplos de uso con Kinect [48].

4.4 Entendiendo el SDK de Kinect. Utilidad práctica.

A pesar de que la información y ayuda para el programador era abundante para la versión anterior de la Kinect, esta nueva versión, debido a los pocos meses que lleva en el mercado,

hace que la información sea escasa. Sin embargo, debido a este motivo, poco a poco van surgiendo diferentes blogs y foros que demuestren la utilidad y funcionalidades de la Kinect con el SDK oficial, ya que la documentación que se proporciona como referencia en [24] es bastante escasa y solo sirve para el aprendizaje de lo más básico. De esta forma, una labor de investigación es necesaria para ver cómo se comporta la Kinect ante diferentes estímulos, y más en concreto, lo que persigue este PFC, detección de UAs en el rostro.

Kinect es capaz de proporcionar información acerca de la posición de la cabeza y la apariencia facial en tiempo real utilizando una máscara en 3D como se puede observar en la Ilustración 11. Esta máscara se crea a partir de un modelo neutro a la que se añaden los parámetros más característicos de la persona bajo seguimiento (tracking). Los datos proporcionados que ofrece Kinect y que permiten construir esta máscara consisten en 3 tipos: puntos 3D (orientación y localización), Animation Units y Shape Units. El espacio de nombres que nos ofrece estas características se encuentra en **Microsoft.Kinect.Face**.

Ilustración 11: Máscara 3D.

El SDK nos proporciona 94 shape units (SUs), definidas en la enumeración FaceShapeDeformations. Cada SU se expresa con un valor numérico que típicamente varía entre -2 y +2, pero que eventualmente puede salirse de este rango. Las SUs son capturas en primera instancia de boca, cejas, y ojos, entre otras partes, para generar el modelo 3D con el que estimar la forma particular de la cabeza del usuario. Además, son constantes mientras dura el seguimiento (tracking). Por tanto, no son objeto de este PFC su estudio.

La versión 2 de Kinect nos proporciona 17 Animations Units, llamadas a partir de ahora K-UAs y que se encuentran definidas en la enumeración FaceShapeAnimation. La versión anterior de Kinect solo proporcionaba 6 Animation Units. Estas K- UAs son un subconjunto de lo que viene definido en el modelo Candide3 (http://www.icg.isy.liu.se/candide/) y son independientes de las SUs. La Ilustración 12 muestra Candide, un modelo parametrizado desarrollado específicamente para la codificación de rostros humanos. Consta de un número relativamente bajo de polígonos (unos 100) que permite una reconstrucción rápida con una potencia de cálculo moderado. Candide se basa en UAs globales y locales. Los globales se corresponden con rotaciones alrededor de los tres ejes. Las locales controlan la “mímica” de la cara para que se puedan obtener diferentes expresiones. Destacar además que, FACS, es una generalización de Candide, al ser este último, creado anteriormente.

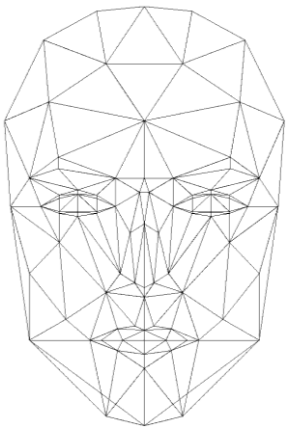


Ilustración 12: Polígonos en Candide (http://www.icg.isy.liu.se/candide/)

En el mundo de la animación y videojuegos, las K-UAs son *deltas* (variaciones) que se obtienen a partir de la forma de la mezcla de los polígonos en estado neutral (como se mostraba en la Ilustración 11), con el fin de transformar los sujetos bajo seguimiento (tracking) en avatares animados para que actúen como lo hace el usuario. En Kinect, cada K-UA se actualiza continuamente cuando el tracking se está realizando (cada frame recibido contiene la información de las Kinect Animation Units).

La mayoría de los K-UAs se expresan con un peso numérico que varía entre 0 y 1. Tres de ellos, Jaw Slide Righ, Right Eyebrow Lowerer, y Left Eyebrow Lowerer, varían entre -1 y +1. JawOpen es la mayoría del tiempo positivo, pero podría darse el caso de que fuera negativo.

Todos los K-UAs tienen valores sin normalizar, lo que indica que una deformación en un K-UA no se tiene que corresponder con la misma escala de deformación en otro K-UA. Además, una sonrisa de la misma magnitud en dos personas diferentes no tiene por qué dar el mismo valor numérico. Recordemos además, que tal y como se explica en la guía de programación para Kinect v2, el número y el significado de cada K-UA está sujeto a cambios y los desarrolladores han comentado en varias ocasiones en los foros oficiales que seguramente este número sea ampliado en versiones posteriores.

En la Tabla 2 se puede consultar un resumen de algunos de las K-UAs proporcionadas más utilizadas con una descripción en forma de avatar, así como los valores que se pueden obtener. Esta tabla se ha obtenido de la guía de programación para Kinect (http://msdn.microsoft.com/en-us/library/jj130970.aspx).

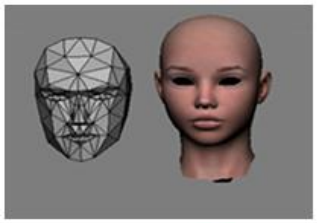
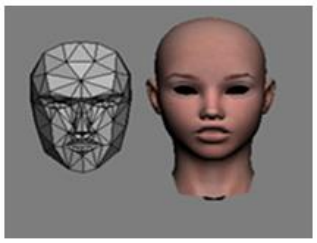
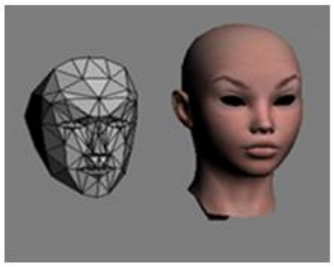
Unidades de Animación	Ilustración	Valor UA
Cara Neutra (todas las UAs son 0)		
UA0 – Inclinación labio superior (en Candide3 es UA 10)		0 = neutros, dientes cubiertos 1 = muestra los dientes completamente -1 = Labio empujado hacia abajo
UA1 – Mandíbula inferior (en Candide3 es UA26/27)		0=cerrada 1=completamente abierta -1= cerrada, como 0
UA2 – Labios (en Candide3 es UA20)		0=neutral 1= totalmente estirado (La sonrisa de joker) -0.5= redondeado (puchero) -1= totalmente redondeado (boca besar)
UA3 – Inclinamiento de cejas (en Candide3 es UA4)		0 = neutral -1 = Levantado casi al máximo + 1 = completamente bajada (hasta el límite de los ojos)
UA4 – Esquinas de los labios (en Candide3 es UA13/15)		0 = neutral -1 = sonrisa (muy feliz) + 1 = muy triste (ceño)
UA5 – Inclinación exterior de las cejas (en Candide3 es UA2)		0 = neutral -1 = Completamente bajada como una cara muy triste + 1 = elevadas como en una expresión de profunda sorpresa

Tabla 2. Animation Units proporcionadas por Kinect

Así mismo, como hemos indicado, Kinect nos ofrece orientación y localización a través de la API. Para el primero, haciendo uso de la propiedad FaceRotationQuaternion (dentro de la clase FaceFrameResult), obtenemos un *cuaternio* que representa la rotación de la cara. Con este cuaternio y el método de Tabla 3 podemos obtener la rotación de la cabeza en los 3 ejes de coordenadas: pitch, yaw y roll. Estos movimientos se muestran en la Ilustración 13.

```
private static void ExtractFaceRotationInDegrees(Vector4 rotQuaternion, out i
nt pitch, out int yaw, out int roll)
{
    double x = rotQuaternion.X;
    double y = rotQuaternion.Y;
    double z = rotQuaternion.Z;
    double w = rotQuaternion.W;

    // convert face rotation quaternion to Euler angles in degrees
    double yawD, pitchD, rollD;
    pitchD = Math.Atan2(2 * ((y * z) + (w * x)), (w * w) - (x * x) -
 (y * y) + (z * z)) / Math.PI * 180.0;
    yawD = Math.Asin(2 * ((w * y) - (x * z))) / Math.PI * 180.0;
    rollD = Math.Atan2(2 * ((x * y) + (w * z)), (w * w) + (x * x) - (y * y) -
 (z * z)) / Math.PI * 180.0;

    // clamp the values to a multiple of the specified increment to control t
he refresh rate
    double FaceRotationIncrementInDegrees = 5.0;
    double increment = FaceRotationIncrementInDegrees;
    pitch = (int)(Math.Floor((pitchD + ((increment / 2.0) * (pitchD > 0 ? 1.0
 : -1.0))) / increment) * increment);
    yaw = (int)(Math.Floor((yawD + ((increment / 2.0) * (yawD > 0 ? 1.0 : -
1.0))) / increment) * increment);
    roll = (int)(Math.Floor((rollD + ((increment / 2.0) * (rollD > 0 ? 1.0 :
-1.0))) / increment) * increment);
}
```

Tabla 3: Cuaternio a grados

Utilizando las conversiones anteriormente comentadas, vemos que los ángulos vienen expresados en grados, con valores que van desde -180 a 180 grados. Internamente, a través de la propiedad floorClipPlane de la clase BodyFrame se obtienen estos ángulos en el sensor en función de la gravedad (relativos a la normal del plano). Por tanto, si el sensor estuviera inclinado, y quisiéramos valores relativos a la horizontal tendríamos que corregir ese offset utilizando la propiedad floorClipPlane para corregirlo. En la Tabla 4 se muestra un resumen de los ángulos y valores que son alcanzables, así como las distintas posiciones necesarias para generarlos.

Ángulo	Valor
Pitch **0 = neutral**	-90 = mirando al suelo +90 = mirando al techo FaceTracking detecta cuando el pitch es menor a 20 grados, pero funciona con más precisión cuando es menor de 10.
Roll **0 = neutral**	-90 = horizontal paralelo con el hombro derecho +90 = horizontal paralelo con el hombro izquierdo FaceTracking detecta cuando el roll es menor de 90 grados, pero funciona con más precisión cuando es menor de 45.
Yaw **0 = neutral**	-90 = girado hacia el hombro derecho +90 = girado hacia el hombro izzquierdo FaceTracking detecta cuando yaw es menor que 45 grados, pero funciona con más precisión cuando es menor que 30.

Tabla 4. Valores de rotación de la cabeza

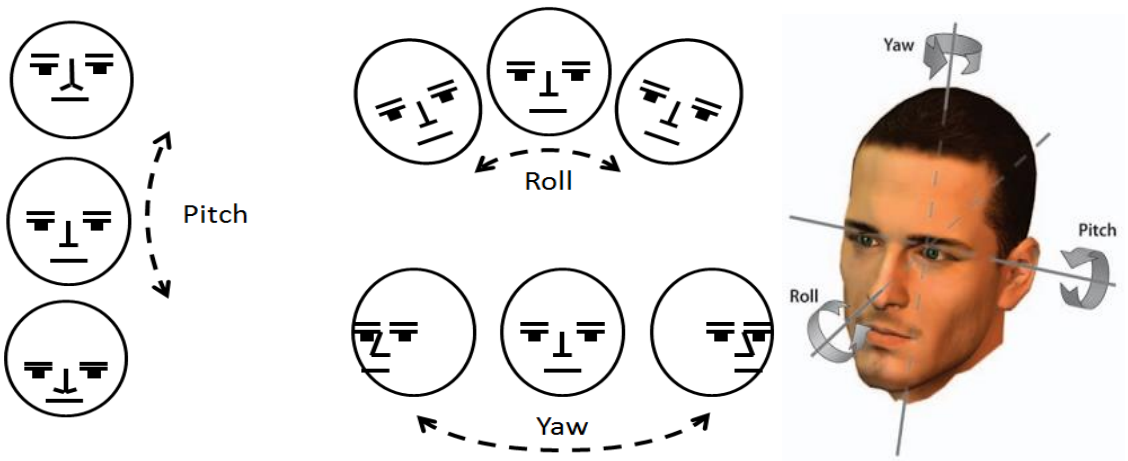


Ilustración 13. Ángulos de giro de la cabeza

Respecto a la localización, para el punto donde se encuentra la cara, Kinect nos ofrece la propiedad HeadPivotPoint. Con esta propiedad, podemos conocer la ubicación de la cabeza relativa al centro óptico de la cámara, que actúa como origen de coordenadas. El eje Z apunta hacia el usuario, el eje Y apunta hacia arriba y el eje X hacia la derecha. La unidad de medida son los metros.

La mayoría de estas K-UAs no se corresponden directamente con las FACS UAs necesarias para reconocer las 6 emociones básicas de las que hemos hablado anteriormente. Por tanto, un análisis para ver la correspondencia que existía fue necesario. Lo estudiamos en el siguiente apartado.

4.5 Kinect y FACS

Para obtener la relación entre las UAs definidas en FACS (Tabla 1) y las K-UAs proporcionadas por Kinect, lo que hemos realizado ha sido grabarnos y analizar los resultados para ver la evolución de las K-UAs más interesantes para nuestro experimento. A su vez, para entender como Kinect funcionaba, se realizó la aplicación básica que se muestra en la Ilustración 14, que permitía mostrar en una misma pantalla la máscara de deformación 3D, una imagen en tiempo real (cámara RGB) y los datos recogidos: las diferentes K-UAs + posición y rotación.

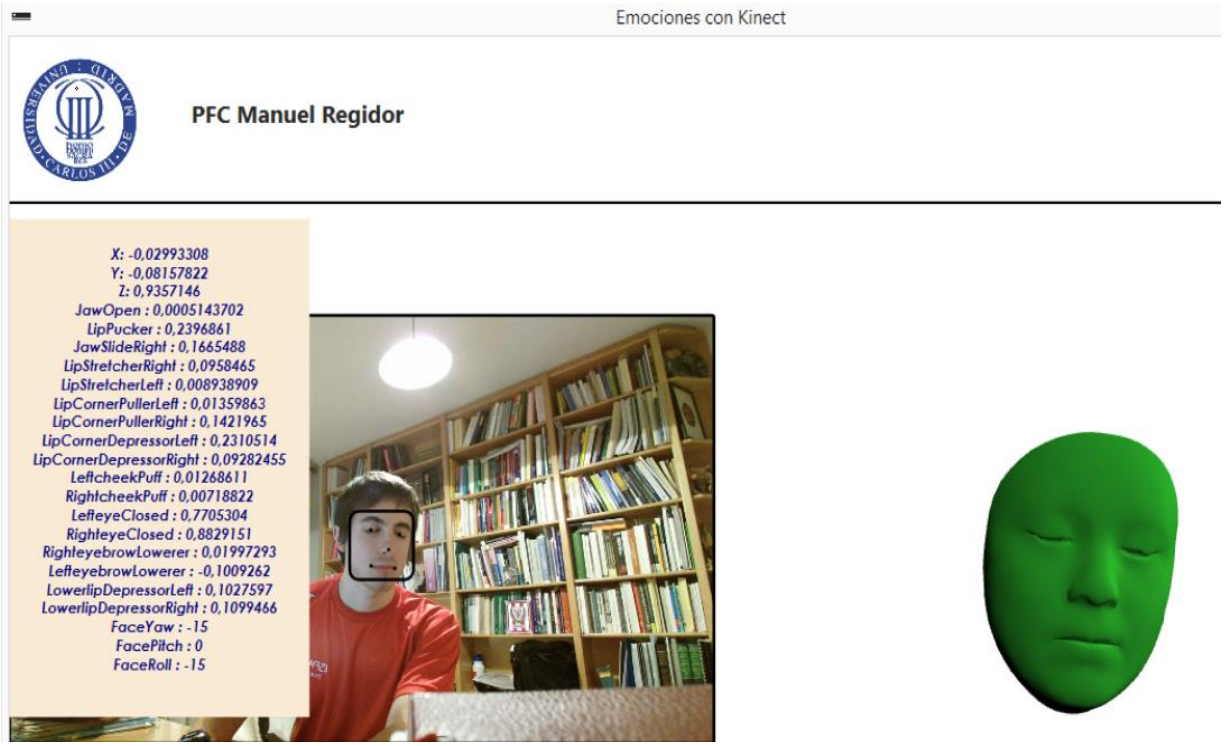


Ilustración 14. Aplicación básica funcionamiento Kinect (I)

En dicha aplicación, con el fin de realizar un seguimiento de la cara, se realiza un encuadre en tiempo de ejecución de la cara con un rectángulo, además, varios puntos son marcados, como los ojos, nariz y boca. Se puede notar como las variables *LeftEyeClosed* y *RightEyedClosed* son valores altos, al ser la captura con los ojos cerrados. También vemos como *JawOpen* es prácticamente un valor despreciable al encontrarse la boca cerrada.

Sin embargo, en la Ilustración 15, habiendo variado los parámetros (forzando el abrir la boca y ojos), se puede observar como las variables descritas anteriormente varían. En este caso, tanto *LeftEyeClosed* y *RightEyeClosed*, son valores muy bajos, del orden de 0.1, mientras que JawOpen es significativamente mayor que en el caso anterior.

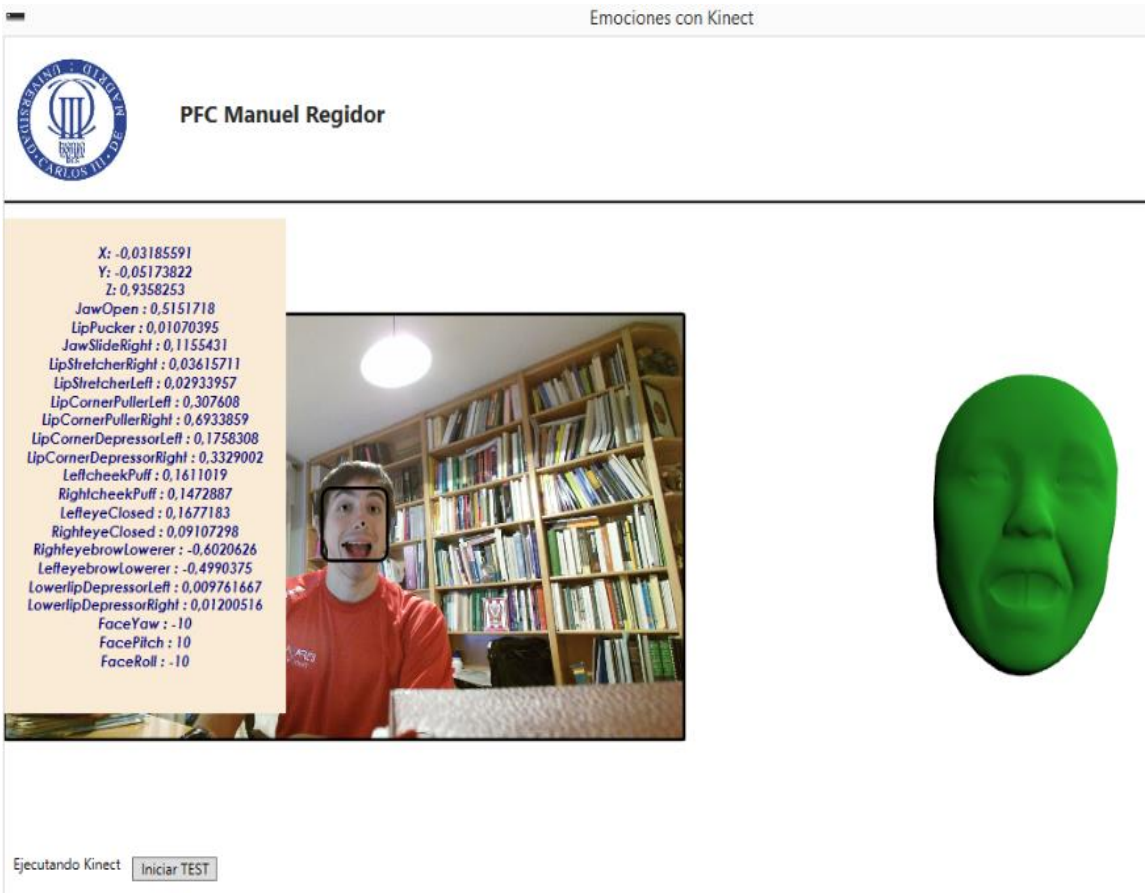


Ilustración 15. Aplicación básica funcionamiento de Kinect (II)

Todo esto nos permitirá realizar un análisis más exhaustivo en los sub apartados siguientes. Empezaremos mencionado la equivalencia entre dos K-UAs: *JawOpen* y *LipPucker* para ver cómo se comportan, para finalmente, mostrar una tabla con la correspondencia de K-UAs y UAs.

4.5.1 Kinect Animation Unit 1: Jaw Open.

Como hemos mencionado anteriormente, basado en la documentación de Kinect SDK, cada K-UA tiene un rango de valores entre 0 y 1. Sin embargo, durante los test realizados, hemos observado que el 0 si se corresponde con la boca (mandíbula) cerrada completamente, mientras que el valor máximo ronda los 0.71, como podemos ver en la Ilustración 16. Esta

ilustración nos muestra la evolución de esta unidad de acción a lo largo del tiempo en tres movimientos de mandíbula que se intentan repetir. JawOpen, se corresponde directamente con la Action Unit 26 (JawDrop, Tabla 1) y describe el movimiento de la mandíbula.

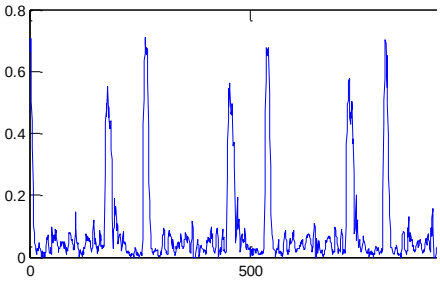


Ilustración 16. Rango de valores para la K-UA JawOpen

4.5.2 Kinect Animation Unit 2: LipPucker.

Esta K-UA se corresponde directamente con la FACS UA 18 (Lip Pucker, Tabla 1). Como hemos hecho antes, hemos probado si los valores que se dictan como referencia en el SDK se corresponden con los obtenidos en las pruebas, y en este caso, el rango es el mismo: entre 0 y 1. En la Ilustración 17 se puede observar el movimiento que Kinect asocia a LipPucker. Los valores máximos se obtienen con la boca en forma de "beso". En la Ilustración 18, se muestra un registro de valores obtenido durante una prueba realizada donde el eje de abscisas se corresponde con el eje de tiempo, en segundos, y el eje de ordenadas nos muestra la intensidad de la mueca. Se puede ver como en lo que dura el experimento, se realizan tres grandes muecas (comienzan en segundo 18, 35 y 50).

Ilustración 17: movimiento asociado a LipPucker (nowcosmetic.co.uk)

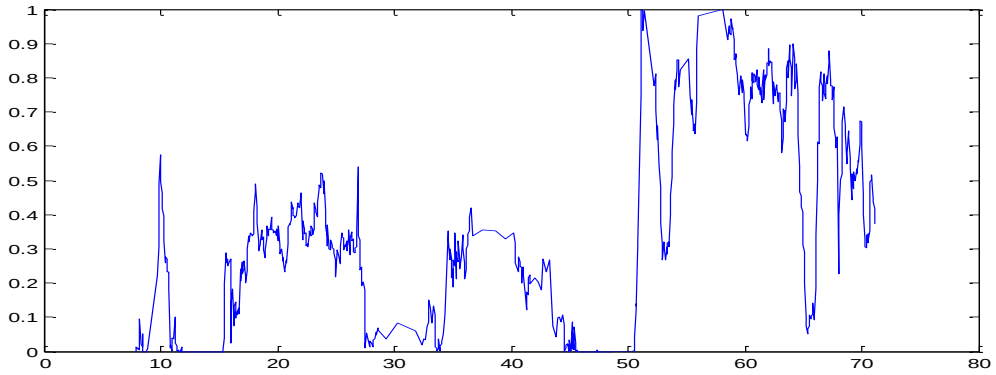


Ilustración 18: rango de valores LipPucker

4.5.3 Otras correspondencias.

Con el resto de K-UAs, gracias al nombre descriptivo que poseen, queda claro a qué movimiento se refiere. CheekPuff, que puede llevar a confusión, se refiere a la forma que se produce cuando se rellenan las mejillas con aire (resoplar). La Tabla 5 muestra una equivalencia parcial entre las Animation Units que son generadas por Kinect, y las Action Units definidas en FACS. Es parcial en el sentido de que Kinect no abarca todas las UAs que existen.

Animation Unit en Kinect	Action Unit (FACS)
JawOpen	JawDrop (Action Unit 26)
LipPucker	Lip Pucker (Actin Unit 18)
JawSlideRight	Jaw Sideways (Action Unit 30)
LipStretcherRight	Lip Corner Puller (AU12) o Lip Tightener (AU23)
LipStretcherLeft	Lip Corner Puller (AU12) o Lip Tightener (AU23)
LipCornerPullerLeft	Lip Corner Puller (Action Unit 12)
LipCornerPullerRight	Lip Corner Puller (Action Unit 12)
LipCornerDepressorLeft	Lip Corner Depressor (Action Unit 15)
LipCornerDepressorRight	Lip Corner Depressor (Action Unit 15)
LeftcheekPuff	[Cheek] Puff (Action Unit 34)
RightcheekPuff	[Cheek] Puff (Action Unit 34)
LefteyeClosed	Eyes Closed (Action Unit 43)
RighteyeClosed	Eyes Closed (Action Unit 43)
RighteyebrowLowerer	Inner Eyebrow Lowerer (Action Unit 42)
LefteyebrowLowerer	Inner Eyebrow Lowerer (Action Unit 42)
LowerlipDepressorLeft	Lips Part (Action Unit 25)
LowerlipDepressorRight	Lips Part (Action Unit 25)

Tabla 5. Animation Units Kinect vs Action Units (FACS)

Estas equivalencias no las proporciona Microsoft, debido a que el número y el significado de las UAs pueden estar sujetos a cambios, y no quieren crear dependencia. Por tanto, para una aplicación en la que se quisiera mayor precisión, por ejemplo, en un clasificador de emociones, un análisis más exhaustivo de los músculos implicados tendría que ser llevado a cabo.

Como veremos en capítulos posteriores, a fin de poder comparar los valores obtenidos de diferentes personas, tendremos que normalizar los valores obtenidos por Kinect.

5. Implementación y diseño de experimentos.

El objetivo principal de este proyecto consiste en el desarrollo de una aplicación mediante Kinect que mostrará una serie de vídeos a un sujeto del cual se obtendrán variables de comportamiento, es decir, respuestas de la persona en forma de movimientos musculares en la cara ante el vídeo que está viendo. Estas variables serán obtenidas a través del análisis facial de la persona analizada en cada segundo del vídeo que está se está reproduciendo. A continuación, describimos primeramente como se desarrolló la aplicación y seguidamente el diseño del experimento que se utilizó para evaluarla.

5.1 Implementación de la aplicación

La aplicación desarrollada en este proyecto se ha implementado utilizando el lenguaje de programación C#, que es muy parecido a Java, que ha sido el lenguaje utilizado principalmente a lo largo de la carrera. El entorno de desarrollo utilizado ha sido Visual Studio 2013 (con licencia del convenio universidad Carlos III de Madrid – Microsoft). Además, se utiliza el lenguaje XAML que permite escribir aplicaciones Windows Presentation Foundation (WPF). XAML es un lenguaje declarativo que simplifica la creación de la interfaz de usuario para una aplicación .NET Framework. Esto es debido a que se separa la creación de elementos visibles en el marcado XAML declarativo de la lógica en tiempo de ejecución mediante archivos de código subyacente. Cuando se representa como texto, los archivos XAML son archivos XML que generalmente tienen extensión .xaml. Los archivos se pueden codificar con cualquier codificación XML, pero lo habitual es la codificación UTF-8.

El programa desarrollado para este proyecto se muestra en el diagrama de flujo de la Ilustración 19. Básicamente, tras la inicialización de los componentes necesarios (InicializarKinect: obtiene sensor Kinect; abre un Reader (lector) para el flujo de datos que queremos conseguir; suscripción a eventos para obtener los datos que van llegando y Open al objeto del sensor), se procede al tracking de la primera persona que aparece delante de la cámara (InicializaFaces). Ésta será la persona seguida en todo momento siempre que no salga del campo visual del sensor (en cuyo caso, se nos avisaría con un mensaje en pantalla: “HDFrame Track lost”). Si por error aparecieran más personas en escena, siempre se seguiría a la primera persona que fue detectada. Después de la inicialización del tracking de la cara, se procede a la inicialización de los componentes necesarios para detectar las K-UAs (InicializaHD).

De esta forma, cada vez que se recibe un evento con un frame HDFace (a través del método highDefinitionFaceFrameReader_FrameArrived), y una vez comprobado que estamos realizando el tracking a una persona y que se está reproduciendo un vídeo en la aplicación, se procede a ir guardando de forma consecutiva los valores de las K-UAs junto con el segundo del vídeo en el que ha ocurrido dicho evento en la variable cadena de caracteres (string) llamada escritura. Dicha variable volcará sus datos en un fichero .txt una vez el vídeo se ha terminado de reproducir con el formato que se muestra en la Ilustración 20. Como se puede observar, tendremos en la primera línea, el segundo del vídeo donde ha ocurrido el evento de recibir el

K-UA y en las sucesivas líneas, cada uno de los K-UAs junto con su valor correspondiente. Todos los valores vienen separados por un punto y coma, de manera que sea más fácil importarlo en MATLAB a posteriori.

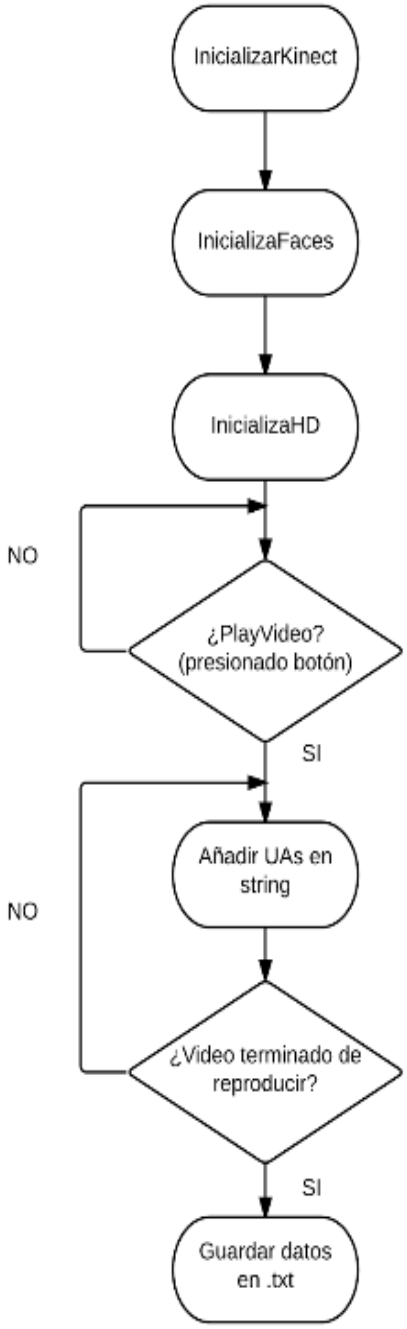


Ilustración 19. Diagrama de flujo de la aplicación desarrollada.

```
Second;7.8661928;7.8866636;7.9364453;7.9952528;8.061924;8.127979;8.1958081;8.
X;-0.001717635;8.043949E-05;0.0002980823;0.0001460422;0.0003348376;-0.0002583
Y;0.1131577;0.1076537;0.1060075;0.1041607;0.1042035;0.1028586;0.1036122;0.104
Z;1.010379;1.005985;1.003692;1.002946;1.002344;1.001646;1.001964;1.002635;1.0
JawOpen;0.4680838;0.5621104;0.5782928;0.5844699;0.6073227;0.5997735;0.6367081
LipPucker;0;0.0004910852;0.01179371;0.006636513;0.01038465;0.004646339;0.0935
JawSlideRight;0.1101471;0.1393798;0.1238204;0.1152312;0.12705;0.1258542;0.127
LipStretcherRight;0;0.004295037;0.005615524;0.004625814;0.002820284;0.0018405
LipStretcherLeft;0.002920736;0.004582732;0.008244555;0.005150215;0.002404963;
LipCornerPullerLeft;0.2457262;0.3718367;0.3793629;0.4000268;0.4347409;0.48412
LipCornerPullerRight;0.6629747;0.88537;0.8851941;0.8728948;0.8400181;0.835622
LipCornerDepressorLeft;0.381865;0.3301672;0.3328923;0.373916;0.4152793;0.2622
LipCornerDepressorRight;1;0.952536;0.9460017;0.952924;0.8969663;0.7148019;0.8
LeftcheekPuff;0.01432045;0.007374158;0.0214683;0.01687578;0.01399333;0.013606
RightcheekPuff;0.3336849;0.05360488;0.02678743;0.02099797;0.02398397;0.022934
LefteyeClosed;0.4088301;0.5968822;0.6038129;0.6264051;0.603268;0.5851651;0.54
RighteyeClosed;0.6587815;0.7783378;0.8218555;0.8804153;0.8076155;0.8342861;0.
RighteyebrowLowerer;0.5510235;0.4158248;0.4422167;0.4722452;0.4450878;0.42942
LefteyebrowLowerer;0.6170354;0.4955778;0.5543944;0.5686101;0.5932314;0.508915
LowerlipDepressorLeft;0.006071241;0.00442329;0.0005478035;0;0;0;0;0;0;0.00055
LowerlipDepressorRight;0.01636022;0.01128398;0.008690882;0.009033609;0.006187
FaceYaw;-15;-15;-15;-15;-10;-15;-15;-15;-15;-15;-15;-15;-15;-15;-15;-15;-10;-
FacePitch;-10;-5;0;0;0;5;0;0;0;0;0;0;0;0;0;-10;-10;-10;-10;-15;-15;-15;-15;
FaceRoll;-5;-5;-5;-5;-5;-5;-5;-5;-10;-10;-10;-10;-5;-5;-5;-5;0;0;0;0;0;5;0
```

Ilustración 20: Fichero con los datos de las K-UAs.

Respecto al apartado visual de la aplicación, con el fin de que el sujeto que estuviera siendo analizado no se sintiera intimidado por datos o por imágenes mostradas (por ejemplo su rostro en una máscara 3D), se simplificó al máximo los datos y gráficas que se muestran quedando tal y como se muestra en la Ilustración 21.

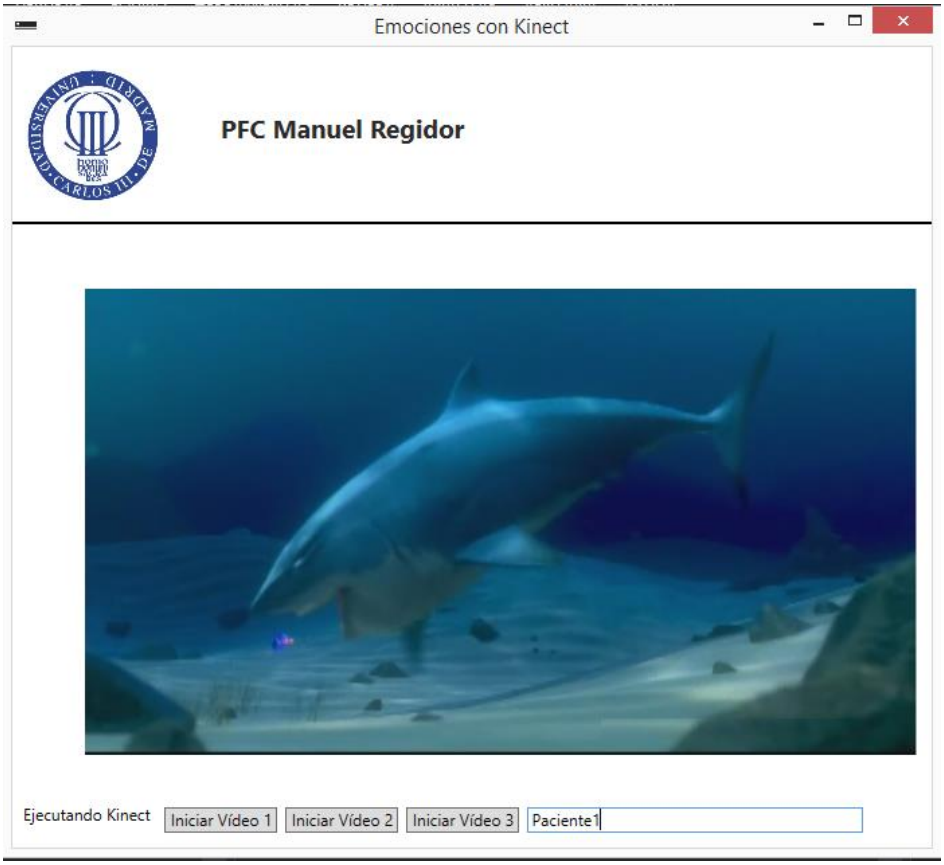


Ilustración 21: funcionamiento de la aplicación

Como se observa en ilustración anterior, la aplicación simplemente consta de un texto que nos indica si se está ejecutando Kinect por si hubiera algún problema y si se está realizando el tracking de la persona (en la imagen, no se realiza tracking, si no pondría “HDFrame Track in progress"). Además, junto a este texto, aparecen tres botones que nos permiten iniciar cada uno de los vídeos de los que consta la aplicación: un vídeo neutro (un tren de Cercanías saliendo de una estación), un vídeo de un corto de Pixar (for the Birds), cuya misión es la de generar diversas emociones y un vídeo, también de Pixar, correspondiente a un fragmento de la película Buscando a Nemo (2003). Estos vídeos serán explicados con mayor detalle en el siguiente capítulo. Finalmente, también disponemos de una caja de texto donde podemos introducir el nombre del sujeto que está siendo grabado a fin de identificar de una manera más sencilla el ‘.txt’ que se generará una vez se acabe de reproducir el vídeo (se genera un ‘.txt’ por cada vídeo reproducido).

5.2 Diseño del experimento.

Durante la realización de este PFC, se han mantenido varias reuniones con el departamento de psiquiatría de la *Fundación Jiménez Díaz (FJD)* (Calle Quintana 11, 28008 Madrid). Este departamento mostró su interés en el proyecto debido a que sus resultados podrían ser utilizados en mejorar las valoraciones del trastorno *por déficit de atención con hiperactividad* (TDAH) en pacientes menores de edad.

Para el diseño del experimento, tal como hemos explicado en la sección anterior, es necesario partir de unos vídeos que vamos a mostrar a los sujetos. En este caso, el primer vídeo por el que hemos optado es un vídeo titulado "*For the Birds"*, que es un corto de animación de Pixar que se estrenó en el año 2000. En [23] se adjunta un enlace donde se puede encontrar más información. El motivo de haber elegido este vídeo es que suele crear diversas emociones en las personas que lo ven. Al principio, parece un vídeo cualquiera de animación, pero a medida que avanza, la evolución de los personajes hace que cause simpatía en el espectador. Esto lleva a una inmediata evolución en los rasgos faciales. De esta manera, como parece evidente, intentaremos buscar patrones comunes de la sensación que causa este vídeo en cada sujeto. El segundo vídeo elegido es un fragmento de la película "Buscando a Nemo" (2003), con el fin de intentar obtener emociones de asombro, susto o sorpresa. Finalmente, el último vídeo elegido fue uno sencillo, que intentara no generar ningún estímulo en la persona que lo está visualizando (vídeo neutro). Se eligió por sencillez un vídeo de un tren de Cercanías saliendo de una estación (obtenido de *Youtube*).

Se realizó un estudio piloto en el que se capturaron datos de 8 sujetos, proporcionados por la FJD: personal administrativo, estudiantes, residentes y médicos, entre otros. El experimento se llevó a cabo en una sala como la mostrada en la Ilustración 22. Una vez explicado el procedimiento, la persona analizada se sitúa a unos 2.5 metros de la televisión, de pie, y limitándose a observar los vídeos. Se recuerda a los entrevistados que ninguna imagen será obtenida de su rostro, y simplemente son datos numéricos lo que se obtiene, a fin de evitar cualquier problema que hiciera tener que usar un formulario de consentimiento. Varios sujetos ya habían visto algunos de los vídeos mostrados, lo que se tendrá en cuenta para su posterior análisis, ya que esto puede hacer que, en determinados momentos, gestos y movimientos que son comunes, no se produzcan.

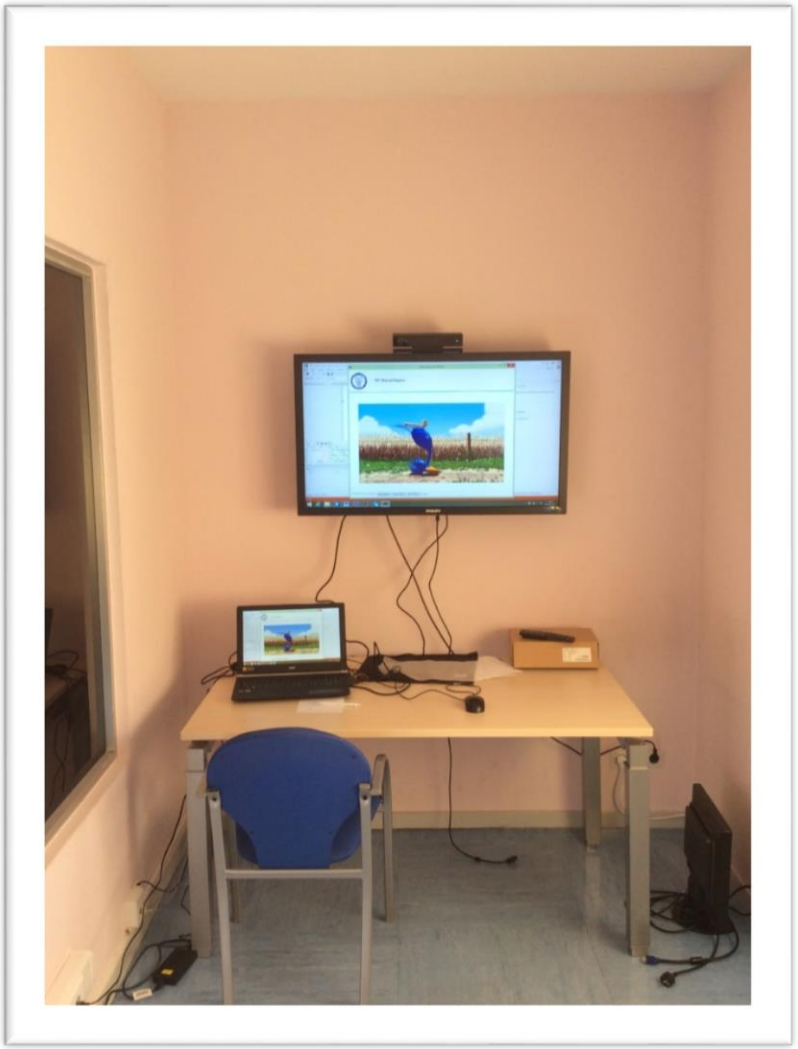


Ilustración 22. Experimento en una sala habilitada de la FJD

6. Resultados

En este capítulo presentamos los resultados del estudio piloto que se realizó en colaboración con el departamento de psiquiatría de la FJD como se explicó en el capítulo anterior. El objetivo es evaluar la fiabilidad y precisión de la cámara Microsoft Kinect y de su SDK en el reconocimiento de diferentes UAs. Este capítulo consta de cuatro secciones. La primera sección explica el tratamiento de datos que se realiza en cada uno de los ficheros generados durante el desarrollo de los experimentos. Las restantes tres secciones se corresponden con el análisis de cada uno de los tres vídeos bajo análisis (tren, pájaros y pez). Es decir, en la segunda sección estudiaremos el comportamiento de los sujetos ante el vídeo del tren, en la que no se espera obtener reacciones agradables ni desagradables al considerarlo un vídeo neutro. Tiene como finalidad el establecer una línea de base con la que comparar la aparición de estados emocionales en los siguientes vídeos. En la tercera sección, se analizará el video pájaros, en la que a priori, es esperable obtener diferentes reacciones. Para finalizar, se estudian las reacciones de los individuos bajo un vídeo que produce una respuesta inmediata causada por algo imprevisto o extraño. Este vídeo pertenece a un fragmento de la película "Buscando a Nemo" (video pez).

6.1 Tratamiento general de datos recogidos por Kinect

Como se comentó en el capítulo anterior, el comportamiento y emociones del individuo que observa cada uno de los videos era recogido a través de diferentes K-UAs que eran proporcionadas por la Kinect y almacenadas en un fichero de texto junto con la información del instante temporal en que se producían. Este fichero es posteriormente importado a MATLAB y utilizado para realizar los diferentes análisis. En esta sección, no hacemos distinción entre que fichero estamos utilizando ya que no es relevante para el procesado de los datos al ser común a los 3 vídeos.

Una vez importado el fichero, lo primero que realizamos es una representación gráfica de los valores de una K-UA cualquiera para ver cómo se comportan los datos capturados. A modo de ejemplo, el gráfico superior de la Ilustración 23 muestra la evolución de una K-UA de uno de los participantes a lo largo de uno de los videos. Como se puede observar en la gráfica, estos datos presentan una gran variabilidad, lo que dificulta su posterior análisis. Para solventar este problema se optó por utilizar un filtro Savitzky-Golay [49], que es un filtro digital que se utiliza frecuentemente en el suavizado de datos. Este filtro incrementa la relación señal a ruido sin distorsionar la señal (mantiene la forma y altura de los picos) mediante un ajuste de mínimos cuadrados locales con aproximación polinómica. Savitzky y Golay (S-G) mostraron que, ajustando un polinomio a un conjunto de muestras de entrada y evaluando posteriormente el polinomio resultante en un único punto dentro del intervalo de aproximación es equivalente a una convolución discreta. En Matlab, utilizamos el filtro S-G de la siguiente forma:

```
señal_suavizada = smooth(señal,span,'sgolay',grado)
```

En la sentencia anterior, Span indica el número de puntos que van a ser utilizados en la aproximación.

Se hicieron varias pruebas con los diferentes parámetros que acepta el filtro S-G en Matlab como se muestra en la Ilustración 23. En la gráfica superior se muestra la señal original, en la gráfica del medio se muestra la señal tras aplicarla el filtro S-G con 155 puntos y grado 2; finalmente, la gráfica inferior muestra el resultado de aplicar a la gráfica origina un filtro G-S con 100 puntos y grado 3. Se seleccionaron como parámetros 155 puntos y grado 2 ya que visualmente observamos que conservaban el patrón de la señal original. La sentencia en Matlab que realiza este filtro con los parámetros escogidos viene dada por:

```
valor_suavizado=smooth(nombreAU, 155,'sgolay', 2);
```

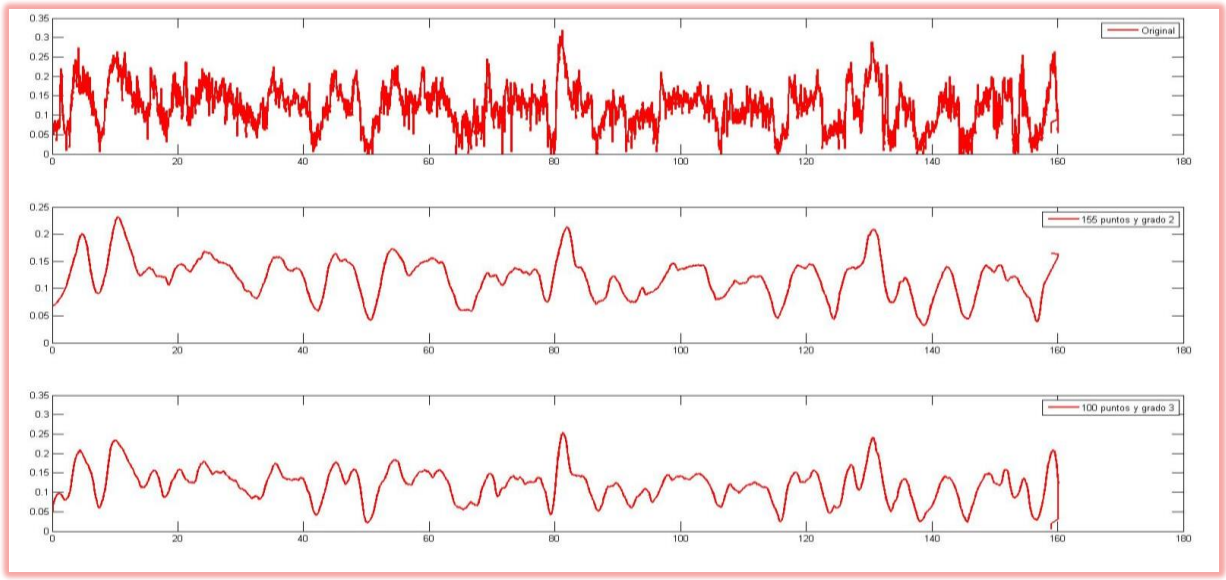


Ilustración 23. Valores de un K-UA obtenido con Kinect

Una vez suavizados los datos para cada K-UA, se procedió a hacer una representación gráfica de cada una de las K-UA de todos los individuos bajo estudio con el objetivo de entender mejor el comportamiento de estas. A modo de ejemplo, la Ilustración 24 muestra la evolución de la K-UA *JawOpen* para cada uno de los individuos.

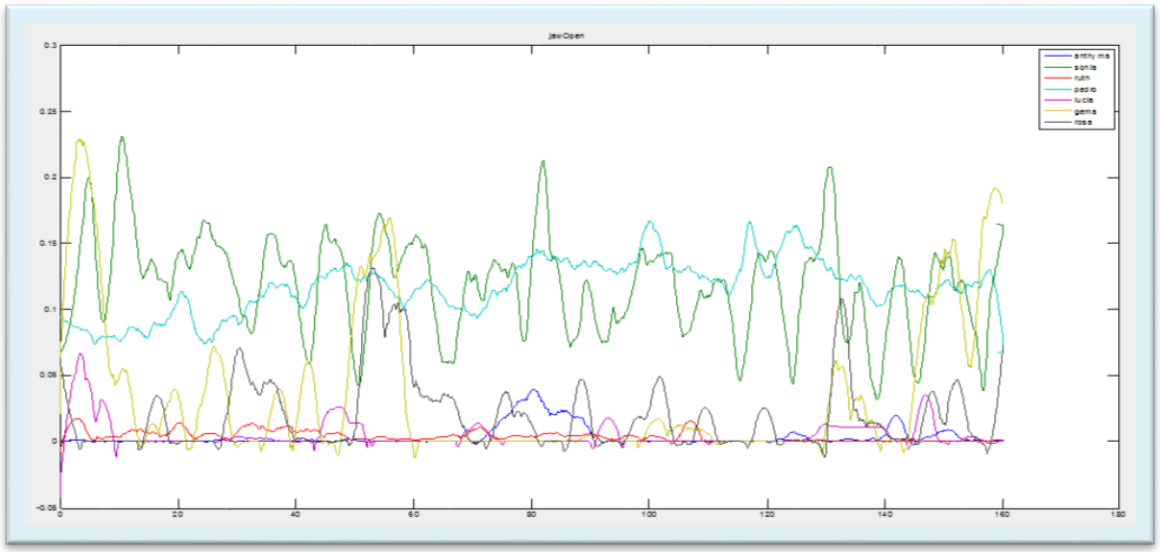


Ilustración 24: K-UA de cada individuo bajo estudio con los valores suavizados

Se puede apreciar que, debido a que los valores no están normalizados, es complicada la comparación entre individuos. Nuestra hipótesis sobre el motivo por el cual los valores no se encuentren normalizados es que estos valores se ven influenciados por la distancia a la que se encuentran los participantes respecto al sensor. De forma específica, nuestra hipótesis es que cuanto más alejado se encuentra el participante respecto al sensor, menor es la precisión de la cámara. Para comprobar esta hipótesis realizamos dos pequeñas pruebas. En la primera, realizamos una grabación en la que con la boca abierta nos vamos alejando del sensor. La Ilustración 25 superior muestra el valor proporcionado por el Kinect SDK del K-UA *JawOpen*, mientras que la gráfica inferior muestra la distancia a la cámara. Se puede apreciar que a medida que nos alejamos, los valores de este K-UA van decayendo en lugar de mantenerse constante como debiera suceder. El segundo experimento consistió en una repetición del anterior pero en este caso la boca se mantenía cerrada. Al igual que con el caso anterior, la Ilustración 26 superior muestra los valores del K-UA *Jawopen* mientras que la gráfica inferior la distancia a la cámara. En este caso, se observa que a medida que nos alejamos esta variable crece en lugar de proporcionar valores cercanos a cero como debería ser. Por tanto, para posteriores estudios, es recomendable realizar una toma de datos con parámetros estrictos de colocación del sensor (a ser posible, cercano a un metro y medio) y respetar estos valores para cada uno de los individuos.

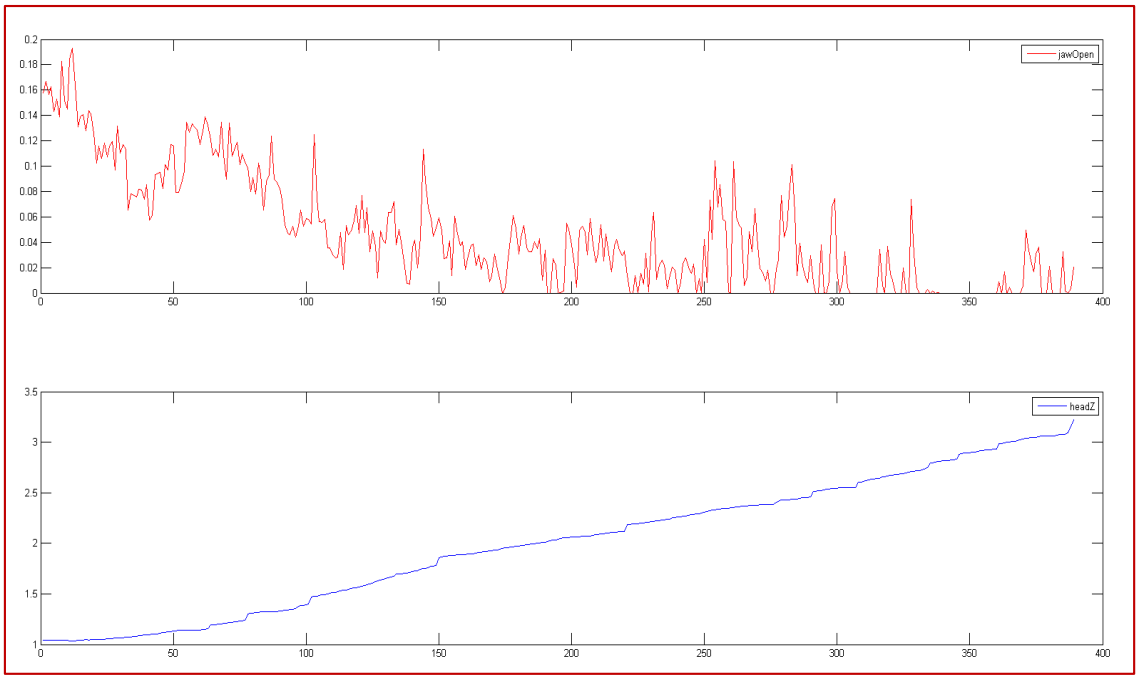


Ilustración 25: experimento consistente en alejarse lentamente del sensor manteniendo la boca abierta.

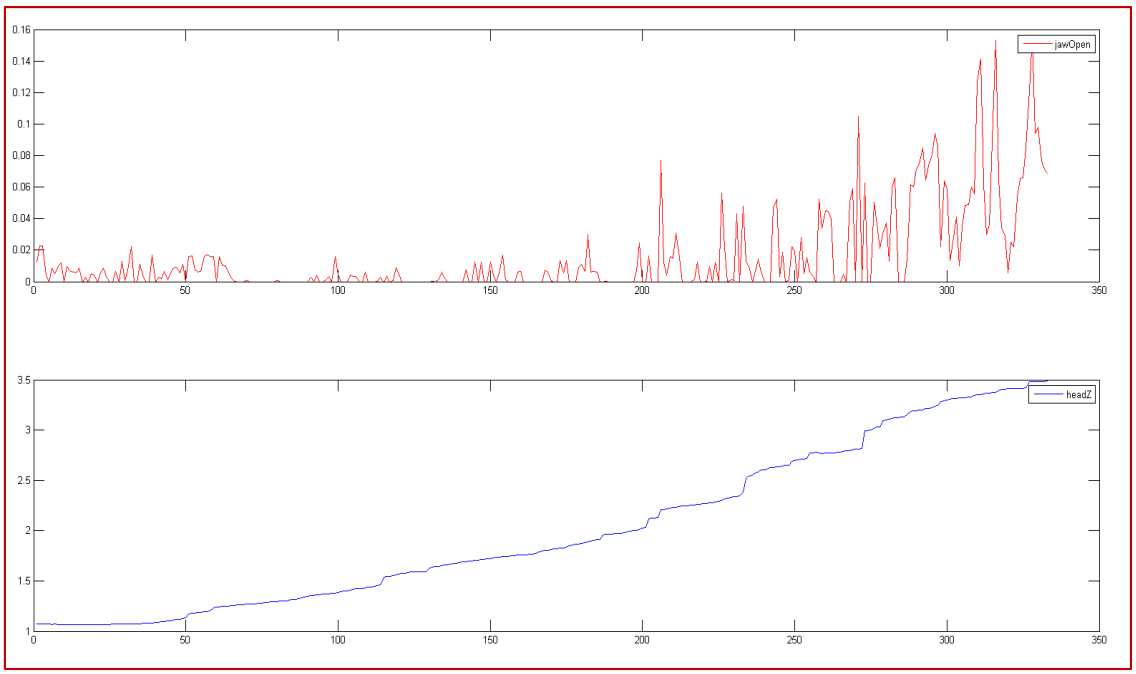


Ilustración 26: experimento consistente en alejarse lentamente del sensor manteniendo la boca cerrada

Como consecuencia de lo expuesto anteriormente, se optó por normalizar los valores utilizando una tipificación (media 0 y desviación 1):

```
valor_normalizado=(valor_suavizado-
mean(valor_suavizado))/(std(valor_suavizado));
```

La Ilustración 27 muestra el resultado de normalizar los datos mostrados anteriormente en la Ilustración 24. En este caso, es más sencilla la comparación de los datos correspondientes a distintos individuos debido a que se encuentran en un mismo marco de referencia. Se puede observar como en la parte central existe un ligero aumento para cada una de los individuos. Este hecho será explicado de forma más detallada en las siguientes secciones.

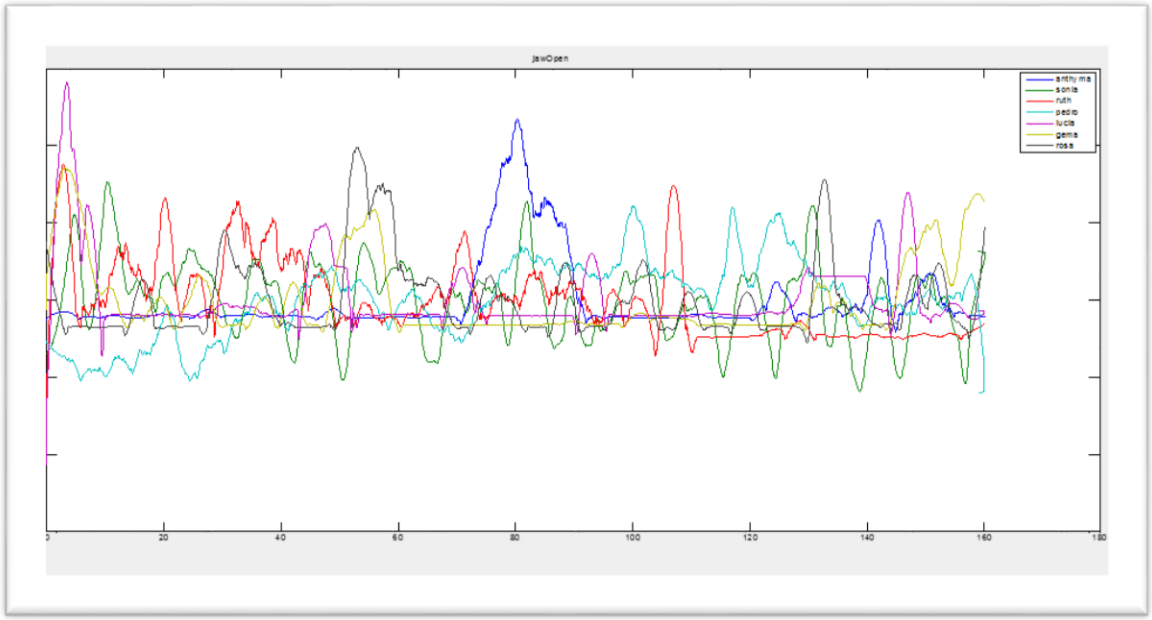


Ilustración 27: K-UA de cada individuo con valores normalizados (y suavizados)

6.2 Primer video (neutro)

Como se ha expuesto anteriormente, el primer vídeo que se mostró a los participantes consistía en la salida de un tren de una estación. Este video es de corta duración y no contiene estímulos agradables ni desagradables. Durante los 37 segundos que dura el vídeo, no se produce ningún hecho importante.

Para tareas de atención visual, tales como las que se pueden dar bajo el presente experimento, la mirada es un claro indicador de si el sujeto está atendiendo al vídeo y de si mantiene cierto grado de concentración. Desafortunadamente, el Kinect SDK no proporciona seguimiento de la mirada. Sin embargo, como hemos visto en capítulos anteriores, ofrece seguimiento de la posición de la cabeza. De esta forma, mediante la variable *FaceYaw*, podemos observar si el sujeto gira la cabeza y no mira a la pantalla donde se reproduce el vídeo (orientada con el sensor Kinect).

Un acontecimiento (no esperado) interesante se produce en los datos capturados cercanos al segundo 25 en los que esta variable toma valores mucho más altos. Esto es consecuencia de que durante estos instantes, el vídeo cambia de plano y el sujeto acompaña al tren con un

movimiento de cabeza. Este hecho se produce también en el resto de los participantes como se muestra en la Ilustración 28. En la Ilustración 29 se muestra la media de esta K-UA para todos los sujetos. Se puede observar, que este acontecimiento se muestra entre los segundos 20 y 35.

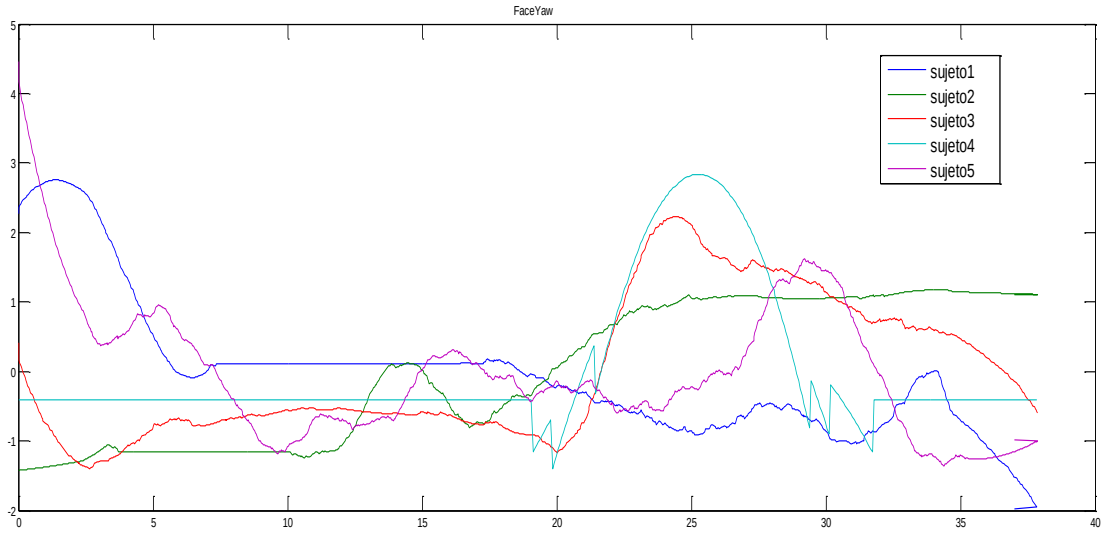


Ilustración 28: K-UA FaceYaw para 5 individuos.

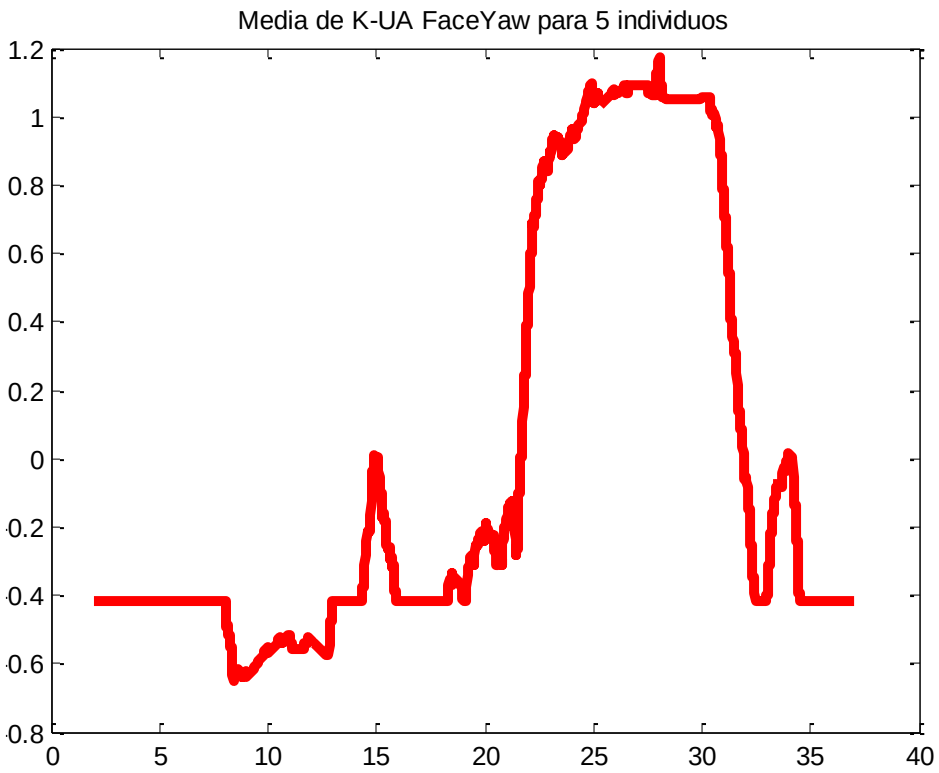


Ilustración 29: Media de FaceYaw para todos los sujetos bajo estudio en el vídeo del tren

Un inconveniente que se encontró al realizar este experimento fue que los individuos no respetaban estrictamente las instrucciones dadas. Esto es, en vez de limitarse a observar el vídeo, muchas veces, por ejemplo, giraban la cabeza preguntando si es normal lo que estaban viendo. De esta forma, es difícil obtener una variable que nos indique el grado de concentración y atención en el vídeo en cada preciso instante. Aun así, podemos obtener la media de la variable para cada individuo.

Participante bajo estudio	*Media FaceYaw (en grados)*
Sujeto 1	-8.0561
Sujeto 2	-15.8793
Sujeto 3	-9.9738
Sujeto 4	6.2437
Sujeto 5	-6.0389
Sujeto 6	12.5811
Sujeto 7	4.7704
Media de los sujetos	**-2.3361**

Se podría considerar que el sujeto está apartando la mirada de la pantalla cuando el valor medio es mayor de 15 grados, según propone Stanley en [52] (relativo al sensor, que en este caso, fue puesto lo más centrado posible). Tanto el sujeto 2 como el sujeto 6 dan valores bastante altos por lo comentado anteriormente, pero en general, el valor medio es bastante cercano al cero, lo que indica que, en este caso, los participantes se encontraban visualizando el vídeo sin distracciones más que las comentadas anteriormente. Esto será importante en un futuro estudiarlo en pacientes con TDAH con el fin de valorar el grado de atención y distracción.

Finalmente, mostramos la evolución de la K-UA *JawOpen* que será de gran utilidad para comparar los otros dos videos. Como sabemos por FACS, la sorpresa se caracteriza por tener las UAs 1+2+5B+26 (ver Tabla 1), donde la UA 26 es *jawDrop*. Representamos esta K-UA junto con los movimientos de los labios en la Ilustración 30. Se puede observar que la variable *JawOpen* (verde), no presenta cambios notables, excepto en el comienzo, lo que puede indicar la sorpresa al encontrarse con un vídeo como el que se visualiza. Además, es importante notar que sus valores son mayoritariamente inferiores a 0,5. Por otro lado, la variable *LipCorner* es característica de la felicidad (UAs 6 + 12), y como vemos, a excepción del principio, también se mantiene constante y dentro del rango -0.5 y 0.5. Por completitud, representamos en azul, *LipStretcher y LipCornerPuller* (tanto derecho como izquierdo). Se puede observar que el recorrido de la variable *LipStretcherLeft* se asemeja al de *LipCornerPullerLeft*. Esta asociación no es tan marcada para las correspondientes dos variables del lado derecho. Esto es probablemente debido a la relación entre la posición del sujeto y el sensor ya el emisor infrarrojo de éste se sitúa en su derecha.

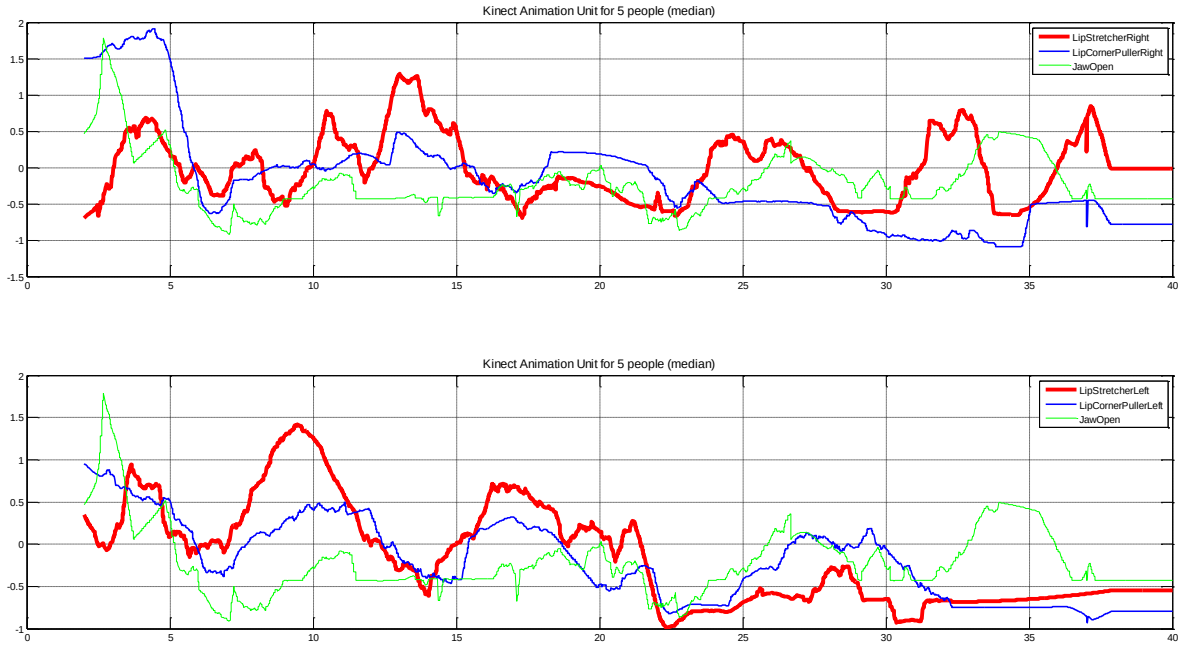


Ilustración 30: K-UAs JawOpen (verde), LIpStretcher (rojo) y LipCornerPuller (azul). En la parte superior, la parte derecha. En la inferior, la izquierda

En vista de los datos presentados, parece evidente que el vídeo escogido para representar un estado emocional neutro ha sido adecuado pues no se han generado grandes cambios en las K-UAs representadas.

6.3 Segundo video (For the birds)

Una vez analizado el vídeo neutro en la sección anterior, pasamos a analizar el vídeo de los pájaros. En esta ocasión, nos encontraremos con reacciones causadas por algo imprevisto, novedoso, extraño y, en principio, gracioso. En consecuencia, esperamos que se produzcan reacciones relevantes en diversas K-UAs.

En esta ocasión, dado que el vídeo bajo análisis va a provocar estímulos de diferente índole en los sujetos, es conveniente tener una herramienta que nos permita relacionar cada fotograma del video con su correspondiente valor de K-UA. Para ello, se desarrolló un script en MATLAB que nos permite mostrar el K-UA de un individuo junto con una ventana en la que se nos muestra el fotograma asociado. El código de este script se puede encontrar en el Anexo III. La Ilustración 31 nos muestra la herramienta desarrollada. Como se puede observar en esta figura, la parte superior muestra el fotograma del vídeo correspondiente al instante que está siendo analizado. Este instante se selecciona mediante un simple clic en la gráfica central o en la inferior en las que se visualiza la evolución de diferentes K-UAs.

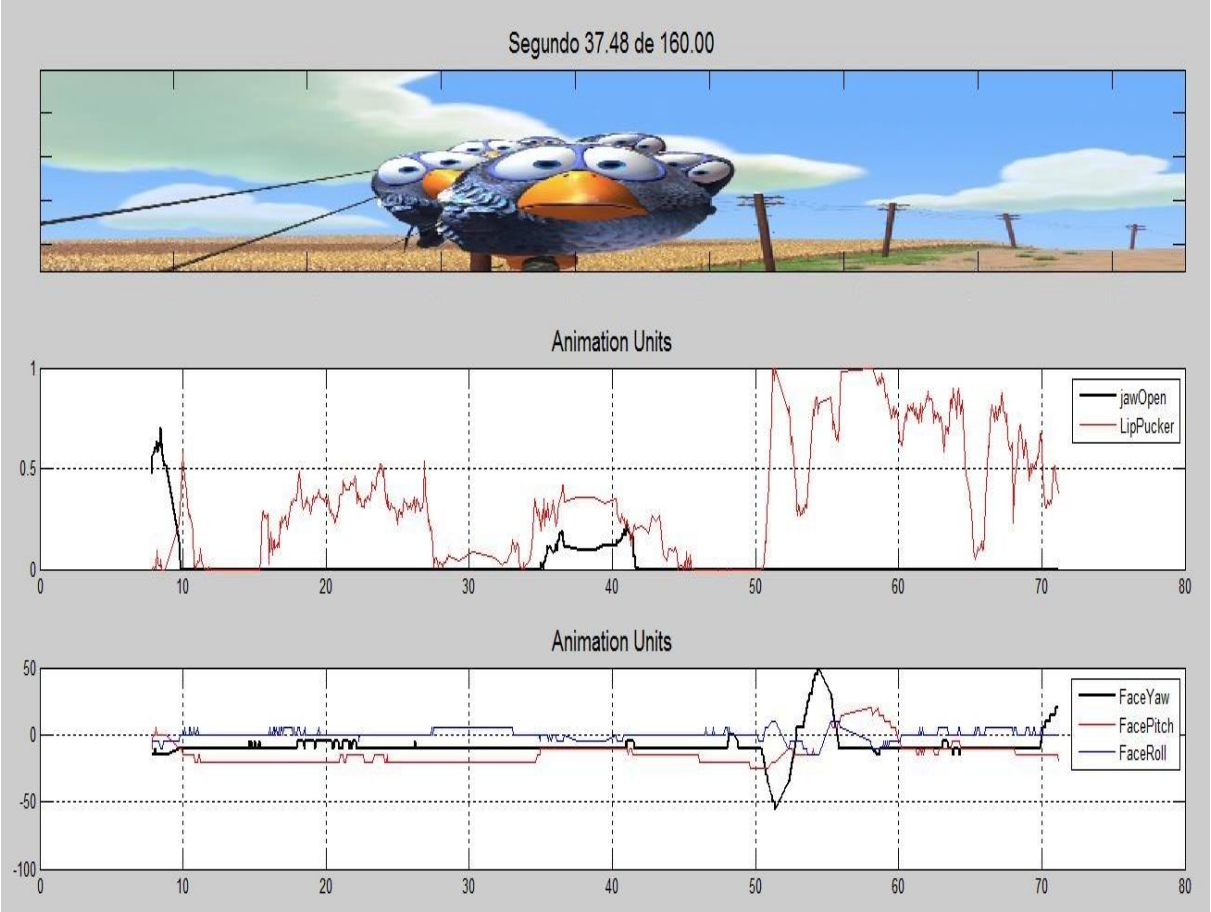


Ilustración 31. K-UAs para uno de los participantes que visualizó el vídeo de los pájaros.

En particular, la ilustración previa muestra dos K-UAs que van a ser importantes durante el análisis de esta sección*: jawOpen* y *LipPucker*. Recordemos que la primera nos indica el grado de apertura de la boca (mandíbula), y la segunda, el movimiento de los labios. Estas dos variables son dos indicadores asociados a la emoción de alegría, que a priori, se espera encontrar dada la naturaleza del vídeo. Como se puede observar, parece haber un máximo sobre el segundo 37 (el eje de abscisas es el tiempo en segundos), momento que se refleja en el fotograma. El fotograma asociado a este segundo es mostrado en la imagen superior de esta ilustración. A continuación, estudiaremos ahora estas dos K-UAs de forma conjunta a todos los participantes para determinar si podemos encontrar patrones similares.

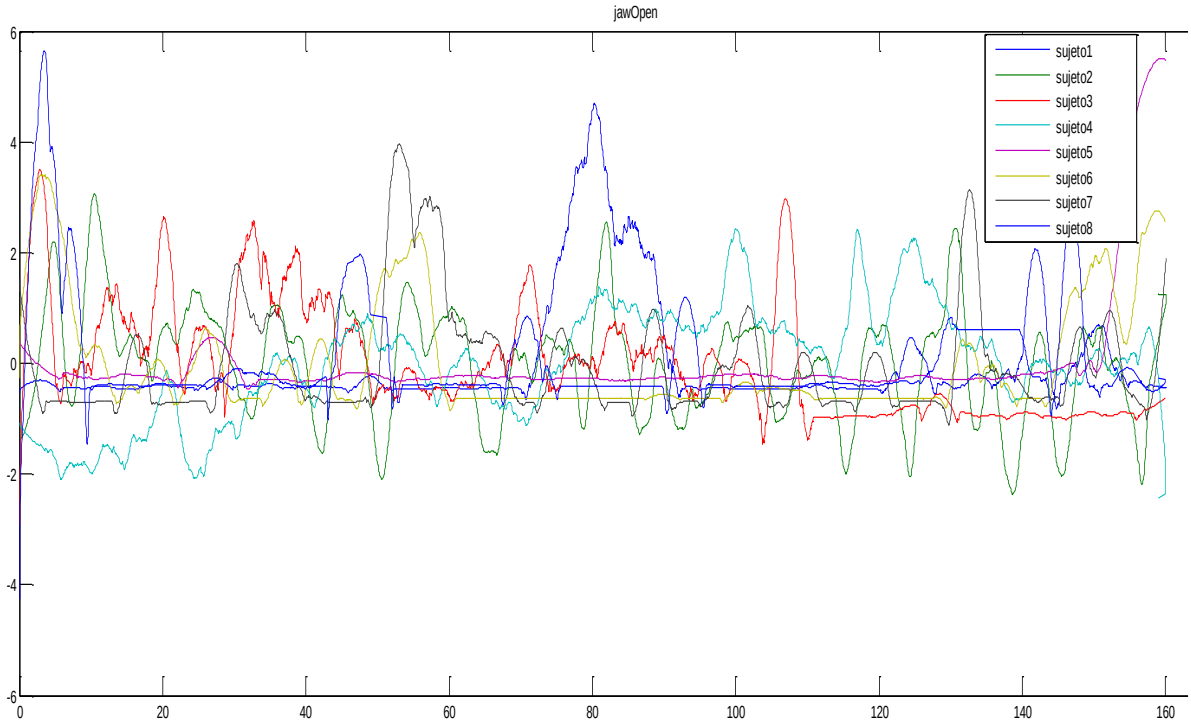


Ilustración 32: jawOpen para el vídeo de los pájaros de todos los sujetos bajo estudio (valores suavizados y normalizados)

La Ilustración 32 muestra la evolución de la variable *jawOpen*, tras ser suavizada y normalizada, para los diferentes participantes. Se puede observan una agrupación de máximos en la zona central (aproximadamente en el segundo 80), como ya adelantamos, así como en el primer y último tercio del vídeo. Para entender mejor la reacción de los diferentes individuos a este segundo video, la media y la mediana de esta variable se muestra en la Ilustración 33. Se opta por representar la mediana ya que de esta forma no se le da tanto peso a variables que son mucho más grandes que el resto (como la variable azul en la Ilustración 32). Podemos notar por tanto, cinco grandes picos que pasamos a analizar con el script anteriormente comentado para ver con qué fotograma se corresponden del vídeo.

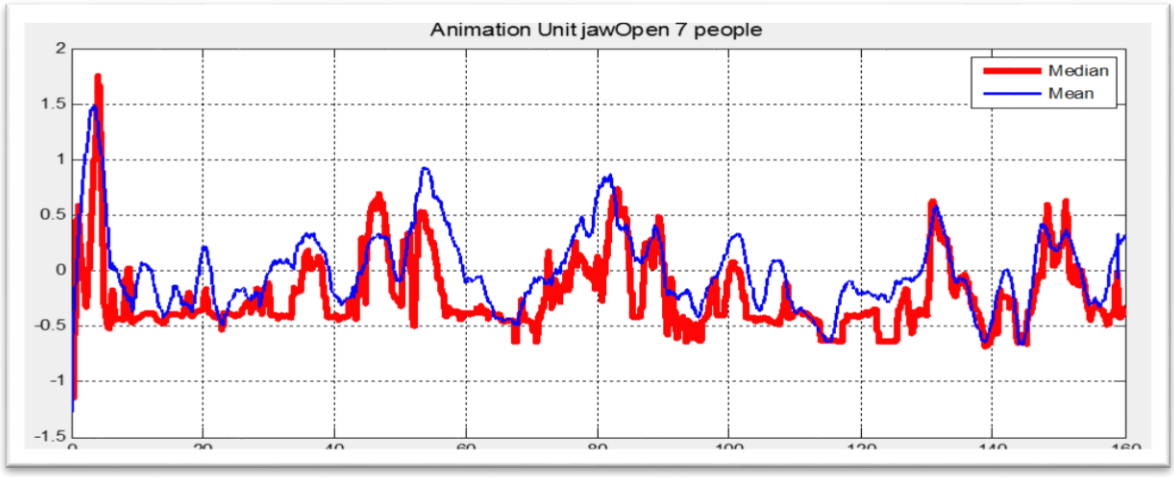


Ilustración 33: media, en azul, y mediana, en rojo, de la variable JawOpen para el vídeo "for the birds" (7 personas). El eje de abscisas es el tiempo del vídeo y el eje de ordenadas la magnitud de la variable jawOpen

El primer máximo (primeros segundos) se corresponde con la simpatía que genera en la persona el reconocer el tipo de video (muchas personas que hicieron el estudio ya habían visto el corto o estaban familiarizados con los videos realizados por esta compañía). A modo de ejemplo, el tercer máximo (situado aproximadamente en el segundo 80), se corresponde con el momento que se muestra en la Ilustración 34: cuando el pájaro grande se posa con los demás pequeños y el cable cede.

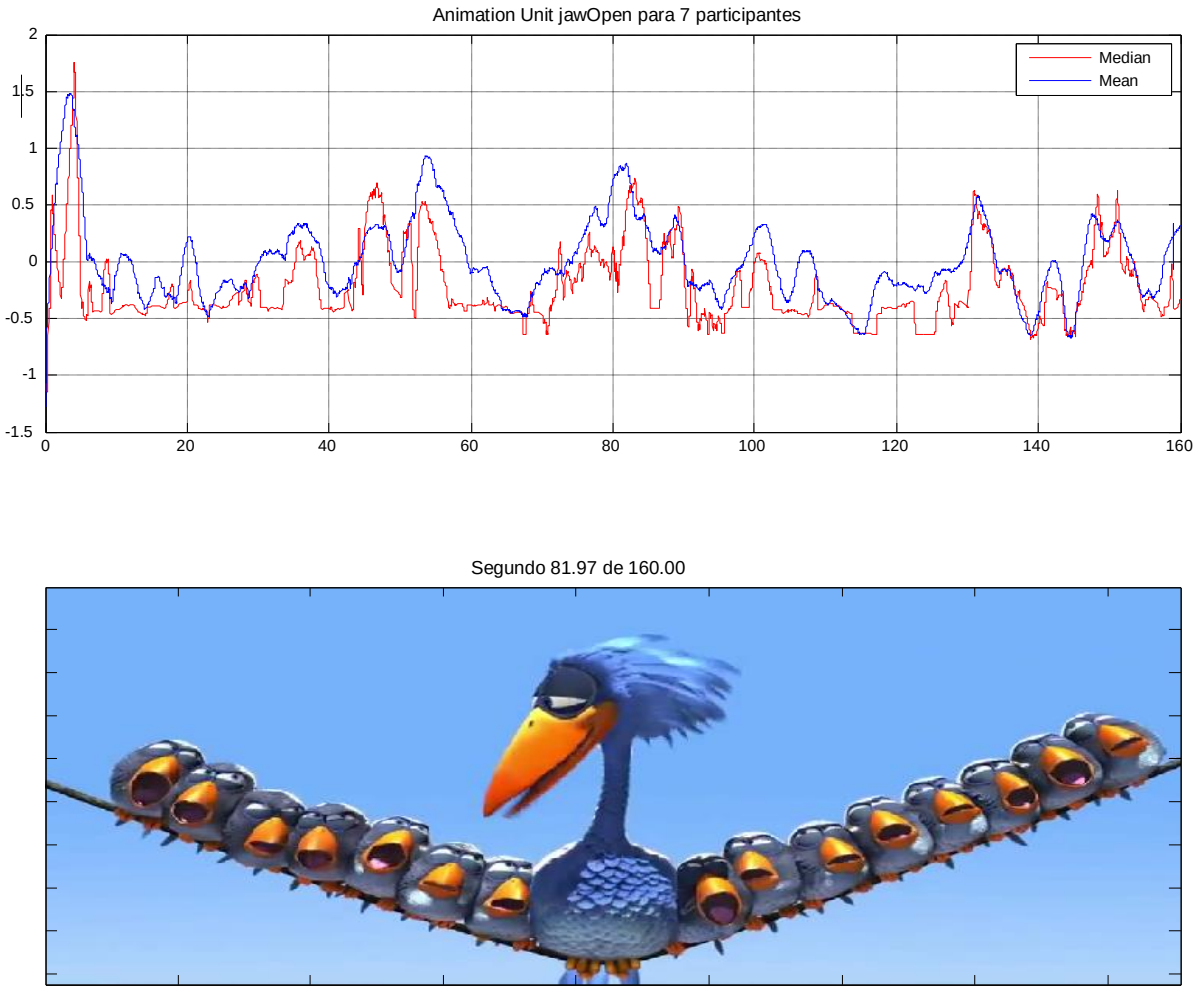


Ilustración 34: fotograma en el segundo 82 de la variable jawOpen con la media y mediana de los datos de todos los sujetos bajo estudio

Por otro lado, sabemos que, en FACS, felicidad se corresponde con la combinación de UAs 6+12 (Cheek Raiser + Lip Corner Puller) (ver Tabla 1). Recordemos que el Kinect SDK nos ofrece la K-UA *LipCornerPuller y CheekPuff*, tanto para el lado izquierdo como derecho. Para entender mejor la reacción de los sujetos a este video, primeramente visualizamos la evolución de estas variables. Además, esperamos encontrar un patrón similar con la variable *jawOpen*.

La Ilustración 35 muestra la mediana de la variable *Lip Corner Puller y CheekPuff* para los distintos sujetos. Además, por motivos de comparación, añadimos inicialmente la variable *jawOpen*. El primer hecho que observamos es que a diferencia del video anterior, los valores son mucho más altos, indicando una mayor reactividad de los individuos. Por otro lado, como habíamos hipotetizado, se observa un patrón similar de las dos variables analizadas con j*awOpen*. En particular, la correlación de *jawOpen* con respecto a *LipCornerLeft* fue de 0.5072 y con *LipCornerRight* de 0.3621. Esta pequeña variación puede ser debida a, como se explicó anteriormente, la relación del sujeto con los sensores de la cámara. Además, se observa que

jawOpen suele seguir a los movimientos originados por LipCornerPuller ya que, al sonreír, primero movemos los labios y después abrir la boca. También se observó que la variable *jawOpen* estaba ligeramente correlacionada con la variable *CheekPuff* (0.3746 y 0.22 para el lado derecho e izquierdo respectivamente). Este hecho tiene sentido ya que esta variable es una aproximación de la variable *Cheek Raiser* anteriormente comentada.

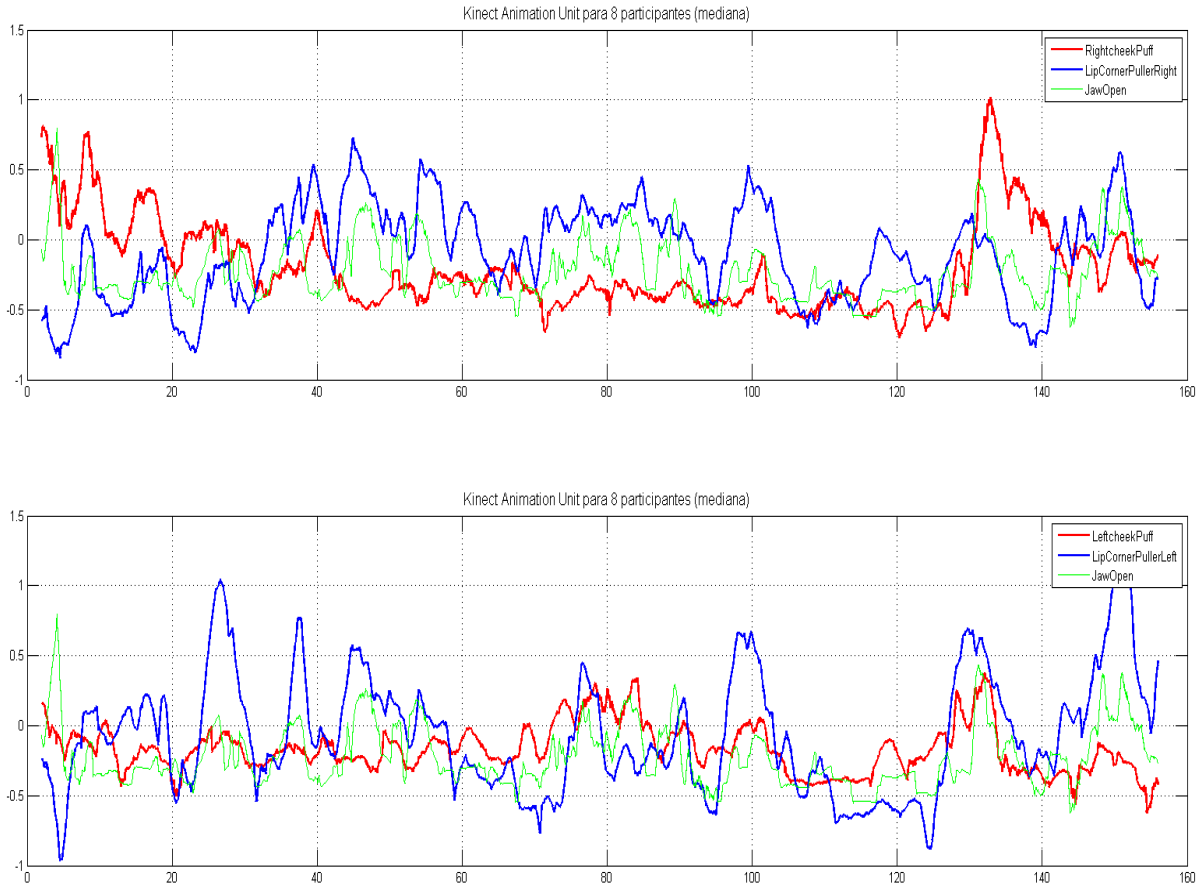


Ilustración 35: Mediana de todos los sujetos bajo estudio para las K-UAs CheekPuff (rojo), LipCornerPuller (azul) y JawOpen (verde). En la parte superior se muestra la parte derecha y en la parte inferior, la izquierda. JawOpen es la misma en los dos casos.

Otro hecho que observamos en la realización del análisis, fue que la variables *LowerLipDepressor*, y *LipCornerPuller* presentaban cierta relación. Este hecho se puede observar en la Ilustración 36 y en el hecho que, por ejemplo, la correlación entre *LowerLipDepressorLeft* y *LipCornerPullerLeft* es de 0.4371.

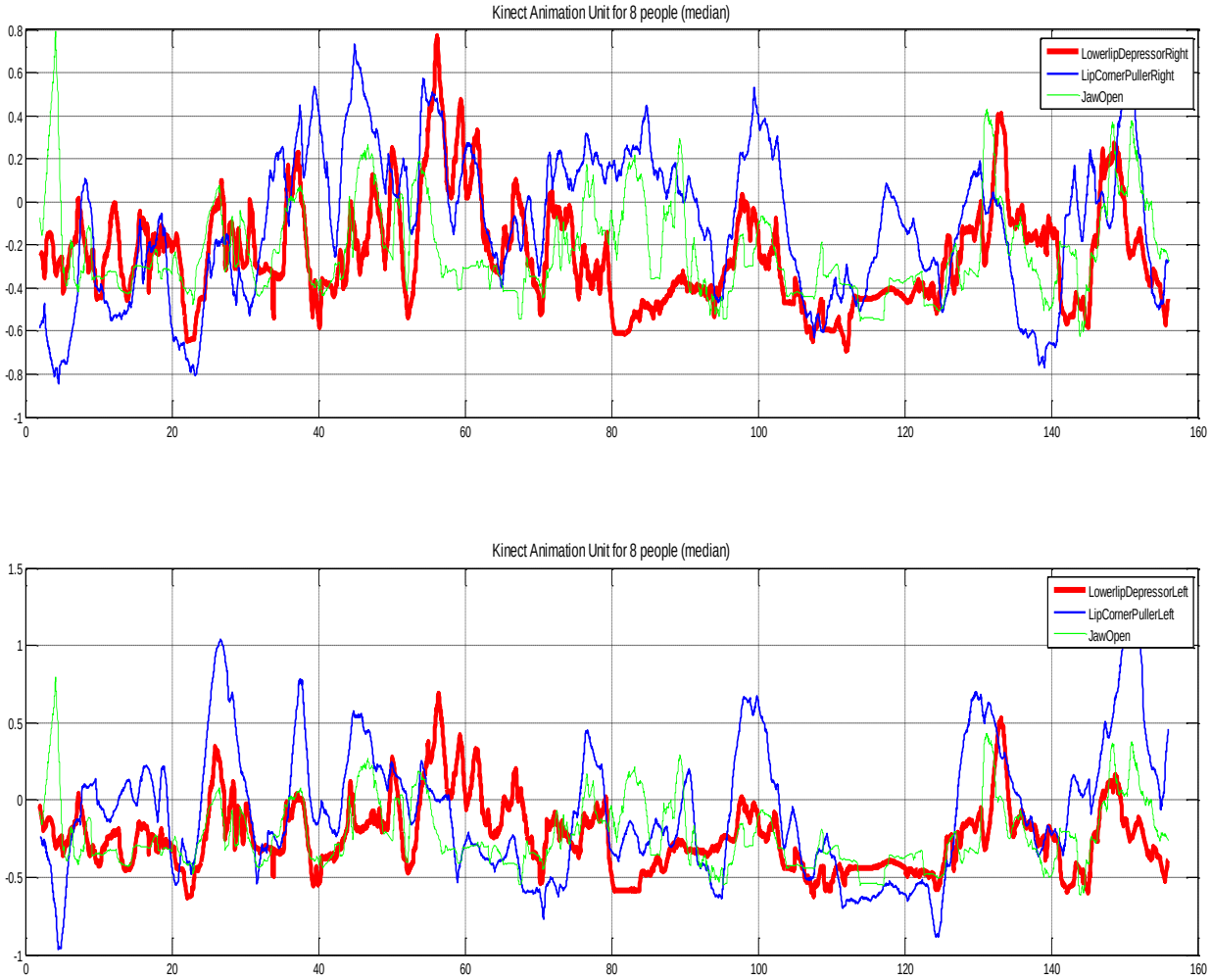


Ilustración 36: Mediana de todos los sujetos bajo estudio para las K-UAs LowerLipDepressor (rojo), LipCornerPuller (azul) y JawOpen (verde). En la parte superior se muestra la parte derecha y en la parte inferior, la izquierda. JawOpen es la misma en los dos casos.

Para concluir esta sección, mostramos una comparativa de la K-UA jawOpen entre el vídeo que se estudió en la sección anterior (tren), y el que acabamos de analizar (pájaros), para cada sujeto.

Participante bajo estudio	Media jawOpen (pájaros)	Media jawOpen (tren)
Sujeto 1	0.0035	-
Sujeto 2	0.0360	0.0044
Sujeto 3	0.0045	-
Sujeto 4	0.0041	6.22E-05
Sujeto 5	0.1169	0.026
Sujeto 6	0.0197	0.005
Sujeto 7	0.0039	-
Sujeto 8	0.1190	0.0182
Media de los sujetos	**0.03845**	**0.0107**

Algunos datos no se encuentran puestos en la tabla anterior pues eran resultados no coherentes como resultado de pérdida de precisión de Kinect. Se puede observar como para el vídeo del tren, el tiempo que la boca permanece abierta en promedio, es menor.

Con los datos que acabamos de mostrar, podemos concluir para este vídeo que gracias a las K-UAs proporcionadas por Kinect y de los conocimientos que nos ofrece FACS, es fácil observar donde se encuentran los momentos que más simpatía generan en el espectador.

6.4 Tercer video (Nemo)

Este tercer video fue seleccionado debido a que, según nuestra consideración, es un video carente de estímulos hasta un determinado punto en donde aparecen de forma repentina y concentrada. Por tanto, esperamos encontrar un número pequeño de máximos en algunas de las K-UAs y de forma muy localizada. En particular, nos vamos a centrar en las emociones sorpresa y miedo. Antes de proceder al análisis, es importante señalar que estas dos emociones tienen a confundirse ya que comparten bastantes UAs. En concreto, se diferencian en la intensidad de la elevación del párpado superior, siempre débil en la sorpresa, mientras que en el miedo puede variar esta intensidad. Recordemos además que la emoción de la sorpresa es frecuentemente seguida por otra emoción que colorea su positividad (sorpresa + alegría) o su negatividad (sorpresa + ira).

Como hemos venido haciendo con los apartados anteriores, pasamos a analizar las variables más relacionadas con estas dos emociones que nos proporciona Kinect. Empezamos mostrando la evolución de los valores de las variables *JawOpen*, *LipStretcher* y *LipCornerPuller* en la Ilustración 37. Estas variables se relacionan con la emoción de sorpresa (en FACS, sorpresa se corresponde con las UAs 1+2+5+26). Se observan dos grandes máximos de la variable *jawOpen* (momento en el cual un pez se encuentra con un tiburón de forma repentina

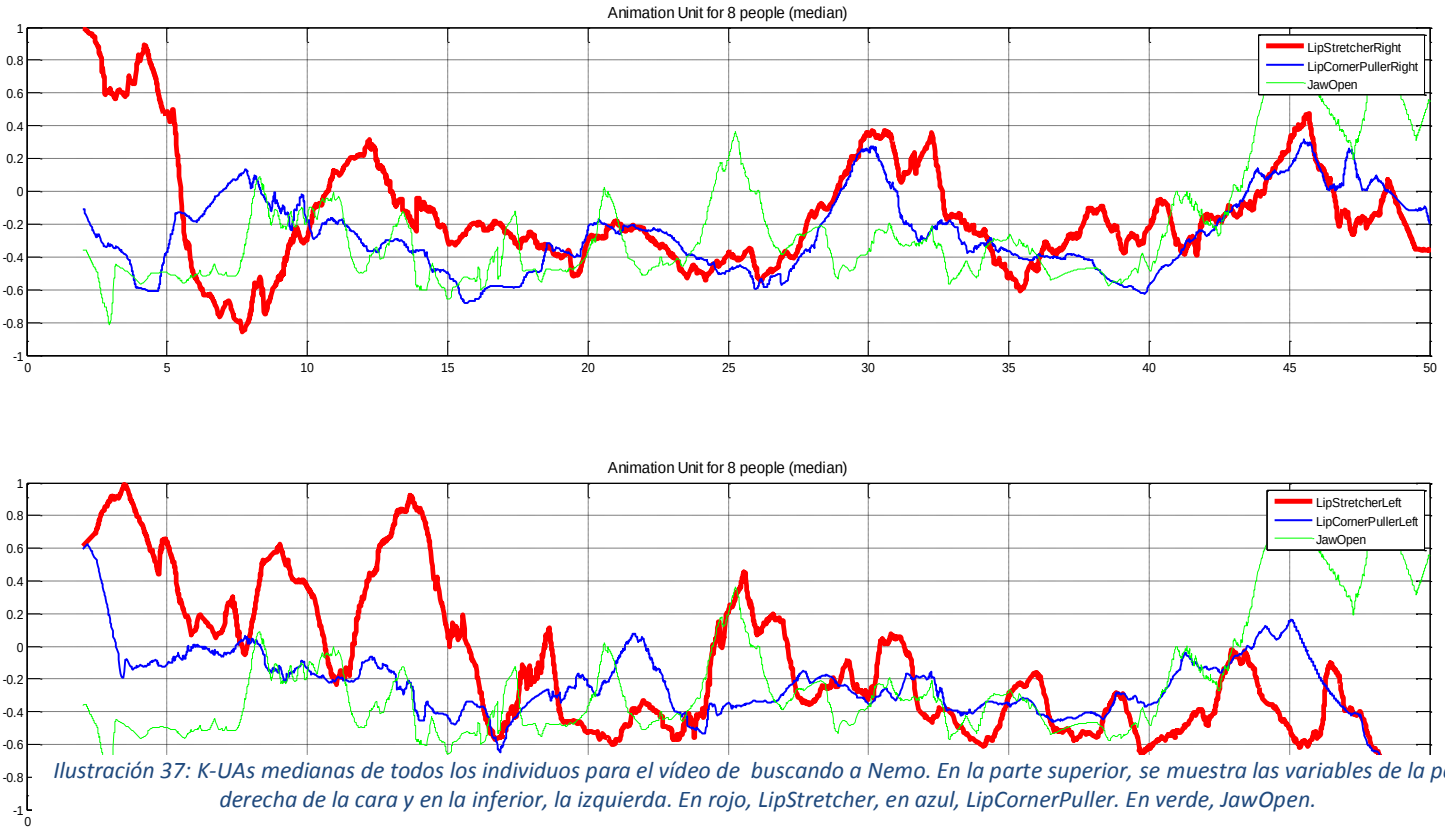


Ilustración 37: K-UAs medianas de todos los individuos para el vídeo de buscando a Nemo. En la parte superior, se muestra las variables de la parte derecha de la cara y en la inferior, la izquierda. En rojo, LipStretcher, en azul, LipCornerPuller. En verde, JawOpen.

cerca del segundo 30 y momento en el que el tiburón comienza a reírse de forma repentina cerca del momento 45) (ver Ilustración 38 e Ilustración 39). También se puede apreciar que, como ocurría en el vídeo los pájaros, los labios acompañan a los movimientos de la boca.

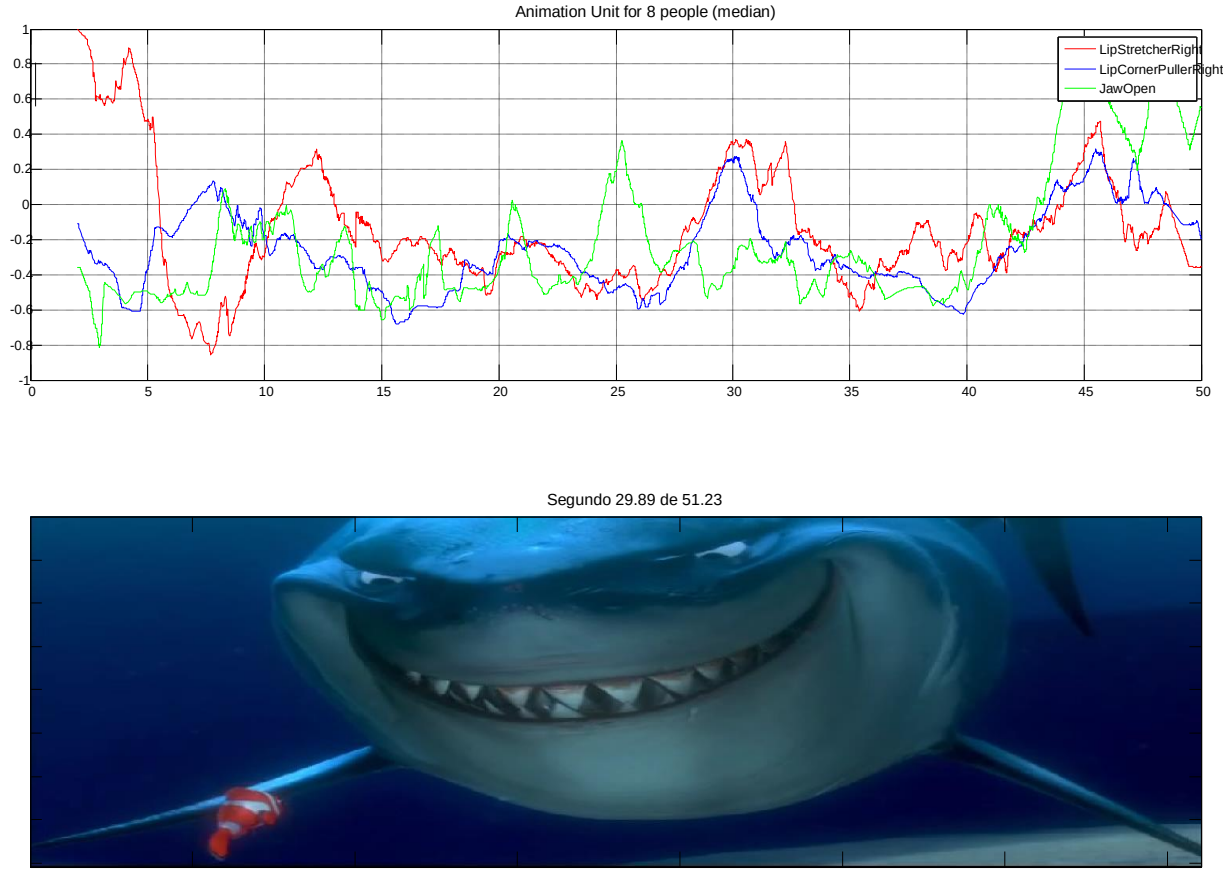


Ilustración 38: segundo 30 del vídeo Buscando a Nemo

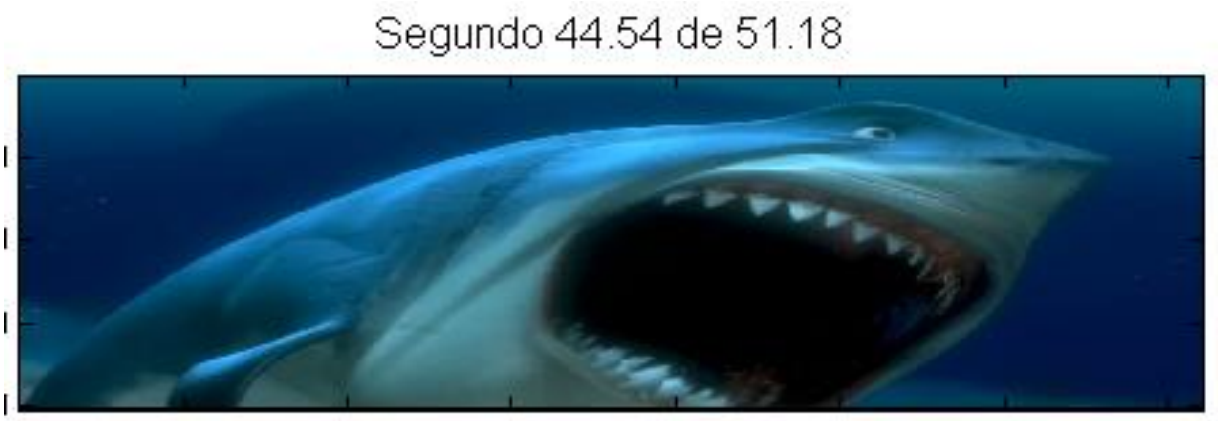


Ilustración 39: segundo 45 del vídeo Buscando a Nemo

En relación a la emoción de miedo, sabemos por FACS, que está relacionado con las UAs 1+2+4+5+7+20+26 (Inner Brow Raiser + Outer Brow Raiser + Brow Lowerer + Upper Lid Raiser

+ Lid Tightener + Lip Stretcher + Jaw Drop). Kinect nos proporciona varias de estas UAs. En la Ilustración 40, mostramos la mediana de la variable *RighteyebrowLowerer*.

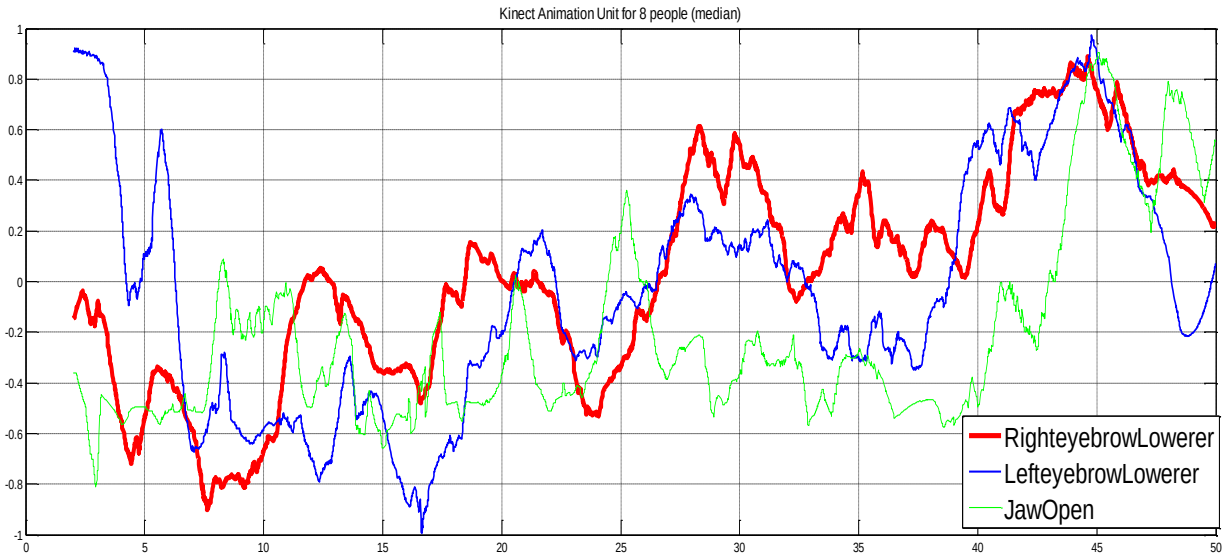


Ilustración 40: Mediana de las K-UAs de todos los individuos para el vídeo de buscando a Nemo. En rojo, movimiento de la ceja derecha, en azul, la ceja izquierda y en verde, la boca.

Se puede apreciar como los movimientos de las cejas (rojo para la ceja derecha y azul para la izquierda) coinciden con los momentos anteriores del vídeo. Este hecho también se cumple para *LipStretcherRight,* como se muestra en la Ilustración 38, donde la variable roja, tiene un patrón de comportamiento muy similar al mostrado por las cejas (correlación 0.21). La variable *LipStretcherLeft* no mostró este patrón. Más análisis serán necesarios en un futuro para obtener un entendimiento más profundo de este comportamiento.

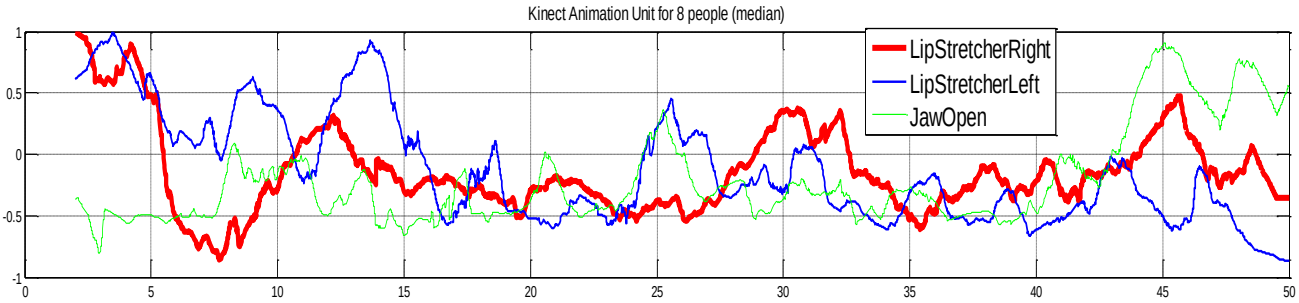


Ilustración 41: K-UA LipStretcher (en rojo para la parte derecha, y en azul, para la izquierda) y K-UA JawOpen para el vídeo de buscando a Nemo. Se muestra la media de los diferentes individuos.

Con estos datos presentados para el vídeo del "pez", queda claro que en los momentos descritos, se ha generado una reacción de sorpresa seguida por miedo, donde el estímulo es el movimiento brusco que hace el tiburón. Es importante destacar que para obtener una reacción válida, el vídeo que se muestra solo debe tener unas pocas situaciones poco comunes, para discernir entre el estado neutro y otras posibles emociones y variaciones de K-UAs.

5. Conclusión y líneas futuras de investigación

En este PFC hemos desarrollado una aplicación para detectar diferentes emociones a través del análisis facial. Esta aplicación tiene gran importancia ya que, actualmente, los análisis automáticos de las expresiones faciales están siendo utilizados, por ejemplo, para apoyar el diagnóstico de enfermedades mentales así como la evaluación de sus tratamientos.

La aplicación ha sido desarrollada utilizando la cámara Kinect que aporta diferentes prestaciones. Entre ellas, su SDK nos da la posibilidad de obtener diferentes indicadores de emociones. Además, el bajo coste de la cámara y la gratuidad de su SDK permiten su fácil adaptación por empresas, centros médicos o particulares.

La robustez de la aplicación ha sido analizada a través de un estudio piloto utilizando la última versión de Kinect (2014, v2). El estudio perseguía identificar las emociones a través de diferentes medidas de comportamiento faciales (K-UAs) proporcionadas por Kinect. Estas emociones eran inducidas a través de la presentación de diferentes videos a los participantes. La versión final de la aplicación es capaz de obtener los distintos indicadores sin presentar fallos. Además, se obtuvieron diferentes resultados de interés. Entre ellos, la capacidad de la aplicación para identificar estados de alegría, miedo o estado neutro. Estos estados fueron identificando utilizando la teoría de las llamada FACS desarrollas por Ekman & Friesen. En particular, se observó que los estados de felicidad eran acompañados de una activación de algunas de las UAs asociadas a labios y mandíbulas. Por otro lado, en el estado de miedo, se producía una activación de UAs asociadas a cejas, comisuras de labios y apertura de la mandíbula. Otra prestación interesante de la aplicación es que los resultados pueden ser obtenidos en tiempo real. El estudio piloto también nos sirvió para identificar distintos aspectos que habrá que cuidar en el futuro como la distancia del sujeto a la cámara. Se observó que si el sujeto se aleja de la cámara la precisión de las mediciones decae considerablemente.

Debido a que el análisis realizado consistió en un estudio piloto y por tanto el tamaño muestral fue pequeño, los trabajos futuros tendrían que ir primeramente encaminados a replicar los resultados obtenidos con una muestra mayor de individuos. Por otro lado, como ha sido comentado anteriormente, la motivación de la aplicación es facilitar los diagnósticos médicos. Por ello, también será interesante analizar el comportamiento de individuos con trastornos psicológicos. En especial, pacientes con TDAH. Finalmente, este estudio piloto se ha centrado en las K-UAs de la cara, pero podría extenderse a la posición del cuerpo y sus movimientos, como por ejemplo, si el participante se encuentra sentado, o de pie, si mueve los pies, la posición de las manos, etc.

Anexo I. Planificación del trabajo

En el proyecto fin de carrera titulado: "reconocimiento de expresiones con Kinect" bajo la dirección del tutor David Delgado Gómez se han estudiado y desarrollado sistemas para un análisis cuantitativo de expresiones faciales ante diferentes vídeos.

Las diferentes fases en las que ha consistido el proyecto son las siguientes:

- Fase I. Documentación y estudio: 51 horas.
  - Estudio del manejo de Kinect a través de video tutoriales (20 horas)
  - Documentación sobre el SDK y lectura de diversos blogs (10 horas)
  - Preparación herramientas de trabajo: instalación de software, compra de material, etc. (6 horas)
  - Pruebas con programas básicos (15 horas)
- Fase II. Desarrollo: 50 horas
  - Diseño de la aplicación en Kinect: 40 horas
  - Implementación y corrección de errores: 10 horas
- Fase III. Experimento y pruebas: 32 horas
  - Pruebas genéricas de la aplicación y correcciones: 10 horas
  - Primeras pruebas con familiares cercanos: 6 horas
  - Pruebas en la universidad Carlos III: 8 horas
  - Toma de datos en FJD: 8 horas
- Fase IV. Análisis: 35 horas.
  - Diseño de un software usando MATLAB para el análisis de los resultados recogidos por Kinect: 15 horas.
  - Interpretación de los resultados: 20 horas
- Fase 5. Memoria: 55 horas
  - Elaboración de la memoria: 40 horas
  - Maquetación: 10 horas
  - Corrección: 5 horas

Fase	Horas empleadas
Fase I. Documentación y estudio	51
Fase II. Desarrollo	50
Fase III. Pruebas	32
Fase IV. Análisis	35
Fase 5. Memoria	55
TOTAL	223

El total requerido para desarrollar este proyecto ha sido de 223 horas.

Anexo II. Costes

A continuación, un desglose de costes de personal, costes del material y costes totales.

Costes de personal

Para la elaboración de este proyecto ha sido necesario el trabajo de un jefe de proyecto, así como el trabajo de un ingeniero.

Puesto	Número de horas	Precio/hora	Importe
Jefe de proyecto	20	90	1.800
Ingeniero	223	60	13.380
TOTAL en euros		**15.180**	

Costes materiales

Concepto	Precio unidad
Asus Eee Top ET2321 (PC todo en uno)	984€
Kinect for Windows v2	200€
Material de oficina	100€
Licencia Matlab	12.000€
TOTAL	13.284€

NOTA: con el fin de minimizar los costes materiales, se puede usar el software libre Octave, que ofrece prácticamente las mismas características que MATLAB.

"El presupuesto total de este proyecto asciende a la cantidad de VEINTIOCHO MIL CUATROCIENTOS SESENTA Y CUATRO EUROS (28.464€)

Leganés a 15 de abril de 2015

El ingeniero proyectista

Fdo. Manuel Regidor Serrano"

Anexo III. Código MATLAB para el análisis

Es necesario que tengamos en la misma ruta que el fichero .m el documento .txt generado por el programa Kinect. En este caso, datos_kinect_0.txt

```
% PFC Manuel Regidor Serrano NIA 100079040 %%%%%%
clc;    % Clear the command window.
close all;  % Close all figures (except those of imtool.)
imtool close all;  % Close all imtool figures.
clear;  % Erase all existing variables.
workspace;  % Make sure the workspace panel is showing.
fontSize = 14;
%lectura de los datos de la Kinect con el separador punto y coma para un partipante
cualquiera
kinect1=dlmread('datos_kinect_0.txt', ';',0,1);
kinect= kinect1(:,1:end-1);



% ¿Image Processing Toolbox installed?
hasIPT = license('test', 'image_toolbox');
if ~hasIPT
    % User does not have the toolbox installed.
    message = sprintf('Sorry, but you do not seem to have the Image Processing
Toolbox.\nDo you want to try to continue anyway?');
    reply = questdlg(message, 'Toolbox missing', 'Yes', 'No', 'Yes');
    if strcmpi(reply, 'No')
        % User said No, so exit.
        return;
    end
end

% Pixar video film (pajaros and nemo are the others)
%folder = fullfile(matlabroot, '.');
movieFullFileName = fullfile('.', 'pixar.mp4');
% Check to see that it exists.
if ~exist(movieFullFileName, 'file')
    strErrorMessage = sprintf('File not found:\n%s\nYou can choose a new one, or
cancel', movieFullFileName);
    response = questdlg(strErrorMessage, 'File not found', 'OK - choose a new
movie.', 'Cancel', 'OK - choose a new movie.');
    if strcmpi(response, 'OK - choose a new movie.')
        [baseFileName, folderName, FilterIndex] = uigetfile('*.avi');
        if ~isequal(baseFileName, 0)
            movieFullFileName = fullfile(folderName, baseFileName);
        else
            return;
        end
    else
        return;
    end
end
```

```
videoObject = VideoReader(movieFullFileName);
% Determine how many frames there are.
numberOfFrames  = videoObject.NumberOfFrames;
frameRate       = videoObject.FrameRate;
duration        = videoObject.Duration;
vidHeight       = videoObject.Height;
vidWidth        = videoObject.Width;

%ANIMATION UNITS
headX=kinect(2,:);
headY=kinect(3,:);
headZ=kinect(4,:);
jawOpen=kinect(5,:);
LipPucker=kinect(6,:);
JawSlideRight=kinect(7,:);
LipStretcherRight=kinect(8,:);
LipStretcherLeft=kinect(9,:);
LipCornerPullerLeft=kinect(10,:);
LipCornerPullerRight=kinect(11,:);
LipCornerDepressorLeft=kinect(12,:);
LipCornerDepressorRight=kinect(13,:);
LeftcheekPuff=kinect(14,:);
RightcheekPuff=kinect(15,:);
LefteyeClosed=kinect(16,:);
RighteyeClosed=kinect(17,:);
RighteyebrowLowerer=kinect(18,:);
LefteyebrowLowerer=kinect(19,:);
LowerlipDepressorLeft=kinect(20,:);
LowerlipDepressorRight=kinect(21,:);
FaceYaw=kinect(22,:);
FacePitch=kinect(23,:);
FaceRoll=kinect(24,:);


% Prepare a figure to show the images in the upper half of the screen.
figure;

% Enlarge figure to full screen.
set(gcf, 'units','normalized','outerposition',[0 0 1 1]);
hPlot = subplot(3, 1, 2);
% Put title back because plot() erases the existing title.

hold off;

plot(kinect(1,:),jawOpen, 'k-', 'LineWidth', 2);
hold on;
plot(kinect(1,:),LipPucker, 'r-');
grid on;
title('Animation Units', 'FontSize', fontSize);
legend('jawOpen','LipPucker');



subplot(3, 1, 3);
plot(kinect(1,:),FaceYaw, 'k-', 'LineWidth', 2);
hold on;
plot(kinect(1,:),FacePitch, 'r-');
hold on;
plot(kinect(1,:),FaceRoll, 'b-');
```

```
grid on;
title('Animation Units', 'FontSize', fontSize);
legend('FaceYaw','FacePitch','FaceRoll');



while 1
    %[x_second,y_second]=ginput(2);
    h = impoint( subplot(3, 1, 2));
    posicion = getPosition(h);
    h.delete;

    x_second=posicion(1);

    lectura= frameRate * x_second;

    if lectura < 1
        lectura= 1;
    end

    if lectura > numberOfFrames
        lectura= numberOfFrames;
    end

    thisFrame = read(videoObject, lectura);
    hImage = subplot(3, 1, 1);
    image(thisFrame);
    caption = sprintf('Segundo %.2f de %.2f', x_second, duration);
    title(caption, 'FontSize', fontSize);


end
```

Referencias

[1] A. Wyrembelsky “Detection of the selected, basic emotions based on face expression using Kinect”, 2013.

[2] Vineetha, Sreeji, Lentin, “Face Expression Detection Using Microsoft Kinect with the Help of Artificial Neural Network”, Trends in Innovative Computing 2012- Intelligent Systems Design: 2012.

[3] Lajevardi, Hussain, “Zernike Moments for Facial Expression Recognition”, International Conference on Communication, Computer and Power 2009: pp. 378-381: 2009

[4] Vretos, Nikolaidios, Pitas, “3D Facial Expression Recognition using Zernike Moments on Depth Images”, Image Processing (ICIP), 2011 18th IEEE International Conference: pp. 773-776:2011

[5] P. Ekman, “Facial Expressions and Emotion”, Am. Psychologist, vol. 48, pp. 384-392, 1993.

[6] P. Ekman and W.V. Friesen, “Pictures of Facial Affect”, Palo Alto, Calif.: Consulting Psychologist, 1976.

[7] P. Ekman and W.V. Friesen. “The Facial Action Coding Systems: A Technique for the Measurement of Facial Movement”. San Francisco: Consulting Psychologists Press, 1978.

[8] P. Ekman. Telling lies: Clues to deceit in the Marketplace, politics, and marriage. New York: WW Norton & Company. 1985/2001

[9] Yan WJ, Wu Q, Liang J, Chen YH, Fu X. “How fast are the leaked facial expressions: the duration of micro-expressions”. J Nonverbal Behav. (2013)

[10] Shreve M, Godavarthy S, Goldgof D, Sarkar S (2011) Macro-and micro-expression spotting in long videos using spatio-temporal strain. 11th Proc Int Conf Autom Face Gesture Recognit (FG2011). Santa Barbara, California IEEE. pp. 51–56.

[11] Pfister T, Li X, Zhao G, Pietikainen M (2011) Recognising spontaneous facial micro-expressions. 2011 Proc IEEE Int Conf Comput Vis (ICCV): IEEE. pp. 1449–1456.

[12] Polikovsky S, Kameda Y, Ohta Y (2009) Facial micro-expressions recognition using high speed camera and 3D-gradient descriptor. 3rd Int Conf on Crime Detection and Prevention (ICDP 2009): IET. pp. 1–6.

[13] Wu Q, Shen X, Fu X (2011) The Machine Knows What You Are Hiding: An Automatic Micro-expression Recognition System. In: D'Mello S, Graesser A, Schuller B, Martin J-C, editors. Affect Comput Intell Interact. Springer Berlin/Heidelberg. pp. 152–162.

[14] Wang SJ, Chen HL, Yan WJ, Chen YH, Fu X (2013) Face Recognition and Micro-expression Recognition Based on Discriminant Tensor Subspace Analysis Plus Extreme Learning Machine. Neural Processing Letters DOI: 10.1007/s11063-013-9288-7

[15] Lucey P, Cohn JF, Kanade T, Saragih J, Ambadar Z, et al. . (2010) The extended Cohn-Kanade dataset (CK+): A complete dataset for action unit and emotion-specified expression. 2011 Conf Comput Vis Pattern Recognit Workshops (CVPRW): IEEE. pp. 94–101.

[16] Aifanti N, Papachristou C, Delopoulos A (2010) The MUG facial expression database. 11th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS): IEEE. pp. 1–4.

[17] Valstar M, Pantic M (2010) Induced disgust, happiness and surprise: an addition to the mmi facial expression database. International Conference on Language Resources and Evaluation, Workshop on EMOTION. pp. 65–70.

[18] Lyons M, Akamatsu S, Kamachi M, Gyoba J (1998) Coding facial expressions with gabor wavelets. 3rd IEEE International Conference on Automatic Face and Gesture Recognition (FG1998) IEEE. pp. 200–205.

[19] Gross R, Matthews I, Cohn J, Kanade T, Baker S (2010) Multi-pie. Image and Vision Computing 28: 807–813 [PMC free article] [PubMed]

[20] Li X, Pfister T, Huang X, Zhao G, Pietikäinen M (2013) A Spontaneous Micro-expression Database: Inducement, Collection and Baseline. 10th Proc Int Conf Autom Face Gesture Recognit (FG2013). Shanghai, China. DOI:10.1109/FG.2013.6553717

[21] Yan, Wen-Jing et al. "CASME II: An Improved Spontaneous Micro-Expression Database and the Baseline Evaluation." Ed. Kun Guo. *PLoS ONE* 9.1 (2014): e86041. *PMC*. Web. 29 Jan. 2015.

[22] K. Scherer and P. Ekman, Handbook of Methods in Nonverbal Behavior Research. Cambridge, UK: Cambridge Univ. Press, 1982.

[23] "For the Birds". Pixar Animation Studios (2000)
http://en.wikipedia.org/wiki/For_the_Birds_%28film%29

[24] The Kinect for Windows SDK https://msdn.microsoft.com/en-us/library/dn799271.aspx

[25] Mase K. "Recognition of Facial Expression from Optical Flow". IEICE Trans E. 1991;74(10):3,474–3,483.

[26] Yacoob Y, Davis L. "Recognizing Human Facial Expressions from Long Image Sequences Using Optical Flow". IEEE Trans Pattern Analysis and Machine Intelligence. 1994 June;16(6):636–642

[27] Rosenblum M, Yacoob Y, Davis L. "Human Expression Recognition from Motion Using a Radial Basis Function Network Architecture". IEEE Trans Neural Networks. 1996;7(5):1,121–1,138.

[28] Lanitis A, Taylor C, Cootes T. “Automatic Interpretation and Coding of Face Images Using Flexible Models”. IEEE Trans Pattern Analysis and Machine Intelligence. 1997 July;19(7):743–756.

[29] Padgett C, Cottrell G. Representing Face Images for Emotion Classification. In: Mozer M, Jordan M, Petsche T, editors. Advances in Neural Information Processing Systems. Vol. 9. Cambridge, Mass: MIT Press; 1997.

[30] Mase K. Recognition of Facial Expression from Optical Flow. IEICE Trans E. 1991;74(10):3,474–3,483.

[31] Terzopoulus D, Waters K. Analysis and Synthesis of Facial Image Sequences Using Physical and Anatomical Models. IEEE Trans Pattern Analysis and Machine Intelligence. 1993;15(6):569–579

[32] Li H, Roivainen P, Forchheimer R. 3-D Motion Estimation in Model-Based Facial Image Coding. IEEE Trans Pattern Analysis and Machine Intelligence. 1993;15(6):545–555.

[33] Himer W, Schneider F, Kost G, Heimann H. Computer-Based Analysis of Facial Action: A New Approach. J Psychophysiology. 1991;5(2):189–195

[34] Kaiser S, Wherle T. Automated Coding of Facial Behavior in Human-Computer Interactions with FACS. J Nonverbal Behavior. 1992;16(2):65–140

[35] Cohn JF, Zlochower AJ, Lien JJ, Wu YT, Kanade T. Automated Face Coding: A Computer-Vision Based Method of Facial Expression Analysis. Psychophysiology. 1999;35(1):35–43

[36] Piana, S., Staglianò, A., Odone, F., Verri, A., & Camurri, A. (2014). Real-time automatic emotion recognition from body gestures. arXiv preprint arXiv:1402.5047.

[37] Szwoch, M. (2013, June). FEEDB: a multimodal database of facial expressions and emotions. In Human System Interaction (HSI), 2013 The 6th International Conference on (pp. 524-531). IEEE.

[38] https://msdn.microsoft.com/en-us/library/jj131023.aspx

[39] Darwin, C. (2002). The expression of the emotions in man and animals. Oxford University Press.

[40] K. Mase, "Recognition of Facial Expression from Optical Flow," *IEICE Trans.,* vol. E74, no. 10, pp. 3474-3483, Oct. 1991.

[41] I.A. Essa and A.P. Pentland, "Coding, Analysis, Interpretation, and Recognition of Facial Expressions," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, no. 7, pp. 757-763, July 1997

[42] M. Bartlett, J. Hager, P. Ekman and T. Sejnowski, "Measuring Facial Expressions by Computer Image Analysis," *Psychophysiology,* vol. 36, pp. 253-264, 1999.

[43] G. Donato, M.S. Bartlett, J.C. Hager, P. Ekman and T.J. Sejnowski, "Classifying Facial Actions," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 21, no. 10, pp. 974-989, Oct. 1999.

[44] http://www.microsoft.com/en-us/kinectforwindows/meetkinect/features.aspx

[45] http://go.microsoft.com/fwlink/?LinkID=513889

[46] Tian, Y. L., Kanade, T., & Cohn, J. F. (2001). Recognizing action units for facial expression analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(2), 97-115.

[47] Stocchi, L. 3D Facial expressions recognition using the Microsoft Kinect.

[48] MVA Jump Start Programming Kinect for Windows v2 videos

[49] Gorry, P. A. (1990). General least-squares smoothing and differentiation by the convolution (Savitzky-Golay) method. *Analytical Chemistry*, *62*(6), 570-573.

[50] Cao, C., Weng, Y., Zhou, S., Tong, Y., & Zhou, K. (2014). Facewarehouse: a 3d facial expression database for visual computing. *Visualization and Computer Graphics, IEEE Transactions on*, *20*(3), 413-425.

[51] Hg, R. I., Jasek, P., Rofidal, C., Nasrollahi, K., Moeslund, T. B., & Tranchet, G. (2012, November). An RGB-D Database Using Microsoft's Kinect for Windows for Face Detection. In *2012 IEEE Eighth International Conference on Signal Image Technology and Internet Based Systems (SITIS),* pp. 42-46.

[52] Stanley, D. (2013). Measuring attention using Microsoft Kinect.

Glosario

AU	Animation Unit; Action Unit
API	Application Programming Interface
ASCII	American Stand Code for Information Interchange
FACS	Facial Action Coding System
FJD	Fundación Jiménez Díaz
IDE	Integrated Development Environment
K-UA	Kinect - Unidad de Animación
NUI	Natural User Interface
PFC	Proyecto Final de Carrera
SVM	Support Vector Machine
UA	Unidad de Acción
ZMs	Zernike Moments